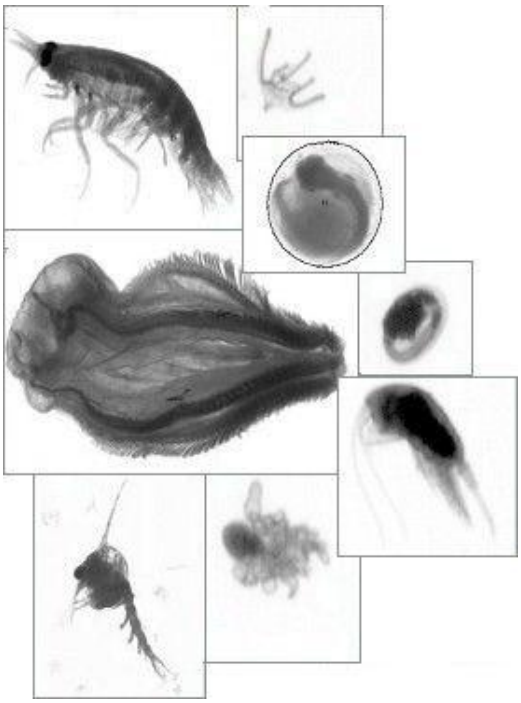
Direction des ressources vivantes
Laboratoire Ressources Halieutiques

Maxime LEDEE
DUT Informatique
IUT Calais Boulogne

Tuteurs Ifremer : Elvire ANTAJAN & Franck COPPIN
Tuteur IUT : Bénédicte TALON

08 avril – 15 juin 2013
Rapport du projet de fin d'études

Modification et amélioration du logiciel *PLANKTON IDENTIFIER*



Remerciements

Je remercie, tout d'abord, Monsieur Dominique Godefroy, Directeur du Centre Ifremer Manche-Mer du Nord, et Monsieur Alain Lefebvre, Chef du Laboratoire Environnement Littoral et Ressources aquacoles, de m'avoir permis de réaliser mon stage au sein du Pôle de Taxinomie et Écologie du Zooplancton (PTÉZoo) de Boulogne-sur-Mer.

Mes remerciements vont également à Madame Elvire Antajan, Chercheur au PTÉZoo, et à Monsieur Franck Coppin, Ingénieur au Laboratoire Ressources Halieutiques de Boulogne-sur-Mer, pour m'avoir accueilli, leur implication dans le suivi de mes missions, leurs précieux conseils et pour m'avoir aidé dans la rédaction de mon rapport de stage.

Je remercie aussi Monsieur Stéphane Gasparini, Chercheur au Laboratoire d'Océanographie de Villefranche-sur-Mer, pour m'avoir guidé dans la compréhension du système *Delphi* et du code source du logiciel *PLANKTON IDENTIFIER*.

Je remercie également Madame Bénédicte Talon, tutrice de stage, pour m'avoir conseillé dans la démarche du projet et dans la rédaction de mon rapport de stage.

Je remercie Madame Valérie Lefebvre, Monsieur Guillaume Wacquet, et Monsieur Wilfried Louis pour leur soutien et leur sympathie.

Je remercie Madame Cyrielle Bandura, stagiaire en Master I FOGEM, pour sa gentillesse et de m'avoir présenté le processus de la préparation des échantillons.

Enfin, un grand merci à l'ensemble du personnel du Centre Manche-Mer du Nord pour leurs aides, leurs conseils et pour m'avoir permis de travailler dans une ambiance agréable et détendue.

Table des matières

Introduction 6

I. Présentation de l'IFREMER 7

1. Présentation générale 7

2. Le centre Manche-Mer du Nord 9

II. Contexte général et objectifs du stage 10

1. Contexte général 10

2. Le *ZooScan* 10

3. Le logiciel *PLANKTON IDENTIFIER* 12

4. Le logiciel *PID VIEWER* 16

III. Définition des besoins 18

1. Les besoins indispensables 18

2. Les besoins optionnels 21

IV. Environnement de travail 22

1. Gestion de projet 22

2. Environnement technique 24

V. Analyse de l'existant 26

1. Le modèles conceptuel de données 26

2. Le modèle opérationnel de traitements 27

3. Représentation physique des données 29

4. Les diagrammes de séquences 32

VI. Développement 37
1. Réalisations indispensables 37
2. Diagrammes de séquences après les modifications 47
3. Réalisations optionnelles 47

VII. Perspectives d'évolutions du logiciel 53
1. Tests unitaires 53
2. Bug focus 53
3. Visualisation des vignettes lors d'une multi-sélection 53
4. Fonction Compilation 53
5. Changement d'IDE 54

VIII. Compétences acquises 55

Conclusions 56

Annexes :

I : Actigramme du processus d'identification d'un échantillon 58
II : Contenu d'un fichier PID d'un échantillon 59
III : Contenu d'un fichier PID « Learn_xxxx.pid » 61
IV : Cahier des charges 62
V : Compte rendu n°3 – Choix de la version d'IDE Delphi 80
VI : Modèle Opérationnel de Traitements des fonctionnalités assurées par le logiciel PLANKTON IDENTIFIER (après modifications) 82
VII : Diagramme de séquences de la fonction Evaluation (après modifications) 83
VIII : Diagramme de séquences de la fonction Prediction (après modifications) 84
IX : Planning effectif 85

Introduction

Ce stage de formation, dans le cadre du DUT Informatique de l'IUT de Calais, a été effectué entre le 08 avril et le 15 juin 2013 dans le Centre d'Ifremer Manche-Mer du Nord à Boulogne-sur-Mer.

Son but était d'accroître mon expérience professionnelle dans le domaine de la programmation informatique. L'objectif de mon stage était de modifier et d'améliorer le logiciel *PLANKTON IDENTIFIER*, conçu pour identifier les objets numérisés d'un échantillon zooplanctonique.

Le rapport est décomposé en plusieurs parties. Je présenterai tout d'abord l'établissement dans lequel il m'a été permis de mener à bien ce stage, l'Institut français de recherche pour l'exploitation de la mer. Je détaillerai ensuite le contexte et les objectifs du stage ainsi que la liste des besoins, l'environnement de travail et les analyses de l'existant. Je présenterai les travaux réalisés puis les perspectives d'évolutions du logiciel. Enfin, je parlerai de mes compétences acquises et j'apporterai mes conclusions.

I. Présentation de l'IFREMER

1. Présentation générale

L'Institut français de recherche pour l'exploitation de la mer (Ifremer) est un établissement public à caractère industriel et commercial, placé sous la tutelle conjointe des ministères de l'Enseignement supérieur et de la Recherche, et de l'Ecologie, du Développement durable et de l'Energie. Né par décret du 5 juin 1984 de la fusion du Centre national d'exploration des océans (CNEXO) et de l'Institut scientifique et technique des pêches maritimes (ISTPM), l'Ifremer est devenu l'établissement le plus intégré en Europe dans le domaine des sciences marines.

1.1 Ses missions

L'Ifremer contribue, par ses travaux et expertises, à la connaissance des océans et de leurs ressources, à la surveillance du milieu marin et du littoral et au développement durable des activités maritimes. À ces fins, il conçoit et met en œuvre des outils d'observation, d'expérimentation et de surveillance et gère des bases de données océanographiques. C'est dans ce cadre qu'il a pour missions :

- Une recherche finalisée afin de répondre aux questions sociétales actuelles (effet du changement climatique, biodiversité marine, prévention des pollutions, qualité des produits de la mer, …). Les résultats couvrent la connaissance scientifique, les innovations technologiques ou les systèmes d'observation et d'exploration de l'océan ;
- La surveillance des mers et du littoral, en soutien à la politique publique de gestion du milieu et des ressources. A partir d'avis ou de rapports d'études, de campagnes d'évaluation, de réseaux de surveillance ou de suivi du milieu marin, l'Ifremer apporte son expertise sur des grandes questions scientifiques dans les domaines de compétences de l'Institut et en lien avec les professionnels ;
- Le développement, la gestion et la mise à disposition de grandes infrastructures de recherche – flotte, moyens de calcul, centre de données, moyens d'essais,

structures expérimentales – à la disposition de la communauté scientifique nationale et européenne, mais aussi dans le cadre de partenariats de recherche public / privé.

1.2 Ses moyens

Pour réaliser l'ensemble des missions qui lui sont confiées, l'Ifremer dispose d'un budget annuel de 214 millions d'euros (hors opérations internes). Il possède 8 navires (4 hauturiers et 4 côtiers), 1 submersible habité, 1 engin téléopéré pour grande profondeur (- 6 000 m) et 2 systèmes sous-marins remorqués.

L'Ifremer emploie environ 1500 chercheurs, ingénieurs, techniciens et administratifs, qui sont répartis dans les cinq centres couvrant l'ensemble des façades maritimes françaises : Manche-Mer du Nord, Bretagne, Atlantique, Méditerranée, Pacifique (fig. 1). Le siège social est situé à Issy-les-Moulineaux.

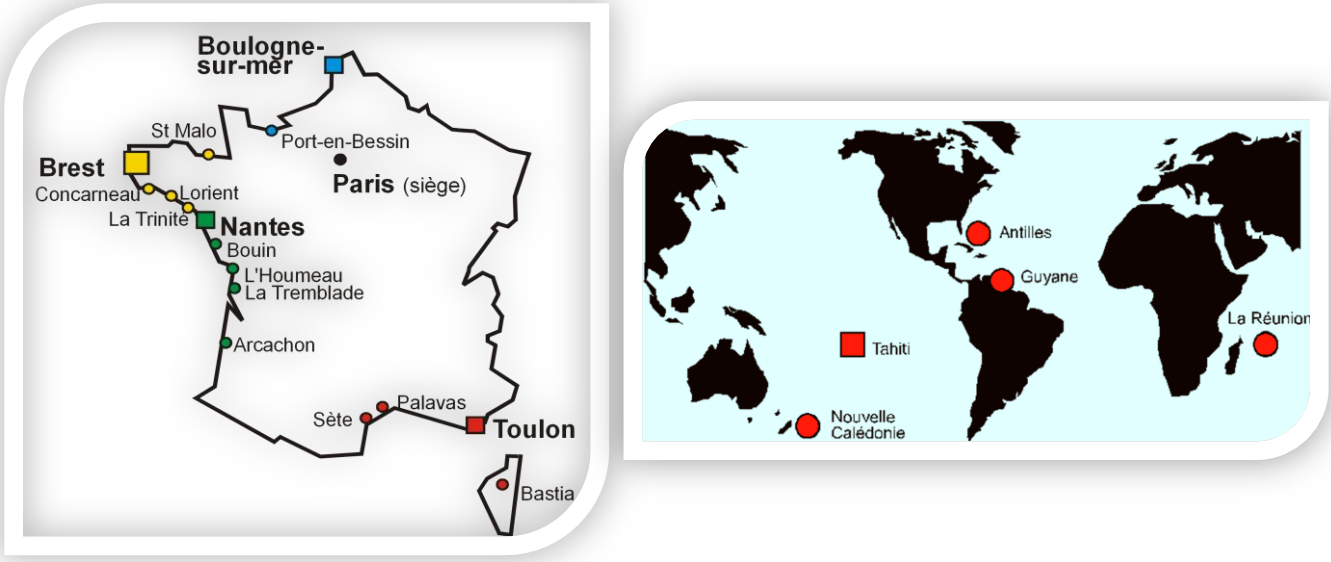


Fig. 1 : Implantation des centres Ifremer (□) et des stations qui leurs sont rattachées (○)

2. Le centre Manche-Mer du Nord

Fig. 2 : Photographie du Centre Manche-mer du Nord à Boulogne-sur-Mer

Situé dans le premier port de pêche français à Boulogne-sur-Mer, le centre Manche-mer du Nord tire sa spécificité de l'importance économique de la pêche et des activités dérivées. Il comporte avec la station rattachée de Port-en-Bessin environ 93 personnes. Son périmètre d'intervention s'étend de la Belgique à la Baie du Mont Saint-Michel, soient environ 960 kilomètres de côtes et couvrent 6 départements et 4 régions. Il est composé d'un service administratif et financier et de 3 laboratoires de recherche :

- Le Laboratoire Ressources Halieutiques, chargé d'améliorer les connaissances sur la biologie, l'écologie et la dynamique des stocks de poissons, et doté d'un pôle de sclérochronologie qui permet d'étudier la croissance des poissons à partir des pièces calcifiées du squelette ;
- Le Laboratoire Comportement des Structures en Mer, chargé de mener des recherches sur l'hydrodynamique des engins sous-marins et des nouveaux concepts offshore, et doté d'un bassin d'essais à houle et courant ;
- Le Laboratoire Environnement Ressources, dont la mission est d'assurer le suivi des réseaux de surveillance de la qualité du milieu marin, et doté d'un Pôle de Taxinomie et Écologie du Zooplancton (PTÉZoo). C'est dans ce laboratoire que j'ai élaboré le cahier des charges avec les maîtres d'ouvrage et que j'ai effectué mon stage.

II. Contexte général et objectifs du stage

1. Contexte général

Chaque année, ce sont plus de 3000 échantillons zooplanctoniques qui sont collectés de manière récurrente par l'Ifremer au cours des campagnes scientifiques de pêche et de surveillance écologique du milieu littoral. L'identification de ces échantillons se fait classiquement à la loupe binoculaire ce qui est très chronophage et requiert des connaissances poussées en taxinomie. En raison de la durée du traitement et du manque d'experts en taxinomie, cette méthode n'est pas satisfaisante pour le traitement d'un grand nombre d'échantillons. C'est pour ces raisons qu'un nouveau système d'imagerie numérique, le *ZooScan*, a été créé au Laboratoire d'Océanographie de Villefranche-sur-Mer (LOV) pour l'identification automatisée des groupes clés du zooplancton (www.zooscan.com). Il combine une technologie avancée de digitalisation avec les algorithmes les plus récents pour l'analyse d'image et le classement des objets. Cette méthode, moins coûteuse en temps d'analyse, permet de traiter assez rapidement un grand nombre d'échantillons, même si elle ne permet pas forcément d'aller aussi loin dans le niveau de détermination taxinomique.

2. Le *ZooScan*

Le *ZooScan* est un scanner permettant la numérisation rapide d'un échantillon liquide de zooplancton contenant des organismes zooplanctoniques fixés (*i.e.* non-vivant ; Fig. 3).

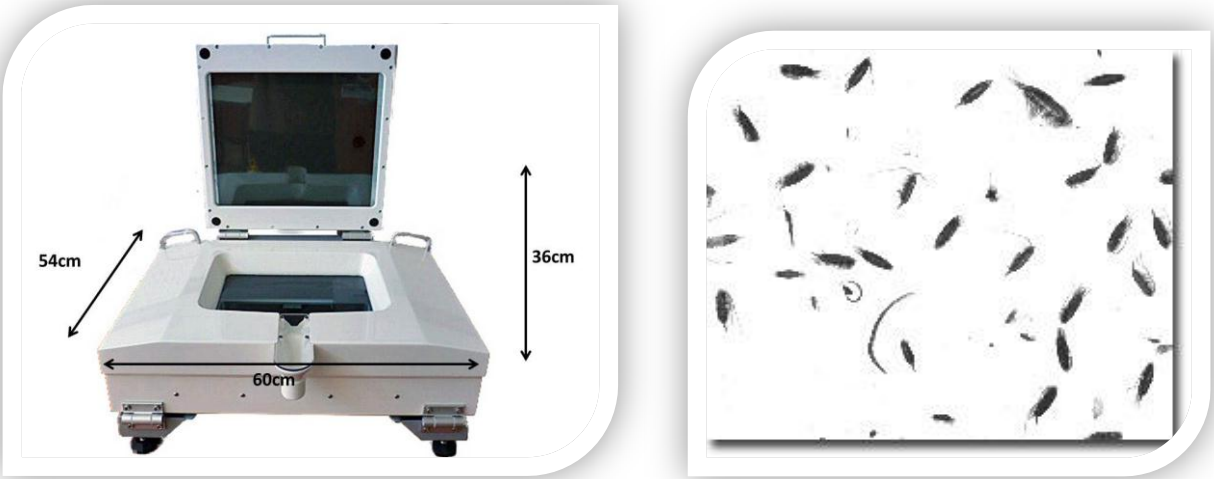


Fig. 3 : Le ZooScan et portion d'une image scannée

Les objets sont automatiquement détectés et leur attributs morphométriques mesurés. Les images de haute résolution (2400 dpi soit 10,583 µm par pixel) acquises sont archivées et le tableau de chaque échantillon scanné contenant les informations pour chacun des objets de l'échantillon est stocké dans un fichier PID nommé « *nom_echantillon*_**dat1.pid** ». Ce fichier peut être immédiatement utilisé dans les différents traitements numériques de données. Les images acquises sont traitées numériquement par le programme *ZOOPROCESS* fonctionnant à partir du logiciel *IMAGEJ*, traitement qui permet d'énumérer et d'individualiser tous les objets de l'image scannée mais aussi de mesurer automatiquement 43 variables morphométriques sur chaque objet. Chaque objet correspond à une vignette sous format **JPEG**. Ces données (le fichier *nom_echantillon*_dat1.pid et les vignettes .jpg) serviront à élaborer les procédures d'apprentissage et d'identification automatique des objets numérisés via le programme *PLANKTON IDENTIFIER*, couplé au logiciel *TANAGRA*.

Le programme *PLANKTON IDENTIFIER* a été conçu en langage orienté objet *Delphi* via l'environnement de développement intégré (IDE) *Borland Developer Studio 2006* par le Dr. Stéphane Gasparini du LOV (http://www.obs-vlfr.fr/~gaspari/Plankton_Identifier/index.php). Ce logiciel utilise en mode batch un logiciel d'exploration de données gratuit, *TANAGRA*, développé par le Dr. Ricco Rakotomalala de l'université Lumière-Lyon 2 (http://eric.univ-lyon2.fr/~ricco/tanagra/index.html).

Le sujet de stage étant la mise à jour et l'amélioration du logiciel *PLANKTON IDENTIFIER*, il s'agira dans un premier temps de faire une présentation de ce logiciel pour ensuite procéder à son analyse afin de pouvoir réaliser les développements prévus au cahier des charges.

3. Le logiciel *PLANKTON IDENTIFIER*

La dernière version utilisée du logiciel *PLANKTON IDENTIFIER* est la 1.3.0. Elle comprend quatre fonctions de base (Fig. 4) :

- *Learning*, pour la création d'un fichier d'apprentissage ;
- *Evaluation*, pour évaluer les performances d'un fichier d'apprentissage ;
- *Prediction*, pour l'identification automatique des objets d'un échantillon ;
- *Validation*, pour visualiser et corriger les résultats de la prédiction.

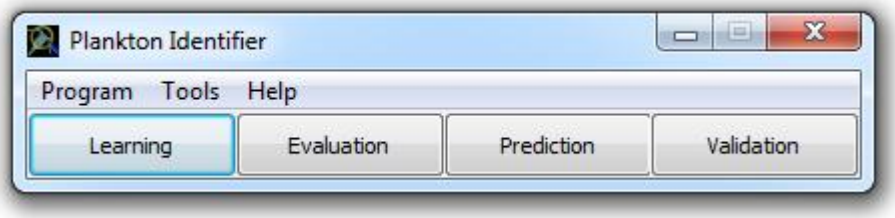


Fig. 4 : Fenêtre principale du logiciel PLANKTON IDENTIFIER

3.1 La fonction Learning

La fonction *Learning* du logiciel permet de créer un ensemble d'apprentissage (*Learning set*) en classant manuellement les vignettes obtenues avec le *ZooScan* dans les groupes taxinomiques définis par l'utilisateur (Fig. 5). Suite à ce classement, le logiciel construit un fichier d'apprentissage nommé « **Learn**_xxxx**.pid** » contenant seulement les mesures morphométriques de chaque objet classé, avec en plus une colonne « **Ident** » référençant le nom du groupe taxinomique d'appartenance et une colonne « Statut » identifiant ces objets comme faisant partie d'un ensemble d'apprentissage (*Learning*).

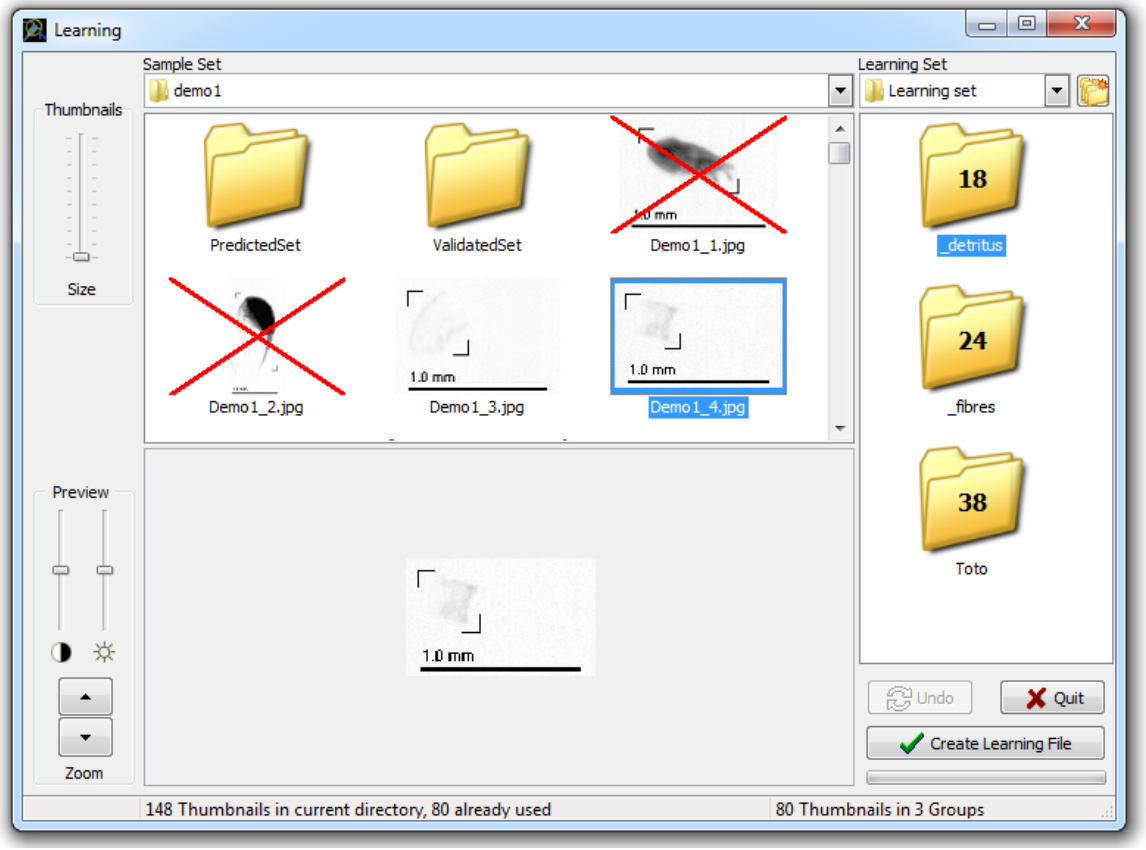


Fig. 5 : Fenêtre Learning

3.2 *Les fonctions* Evaluation *et* Prediction

Le fichier d'apprentissage (Learn_xxxx.pid) créé va nous permettre de construire un modèle de classification à partir de l'une des sept méthodes d'apprentissage supervisé implémentées dans les fonctions *Evaluation* et *Prediction* de *PLANKTON IDENTIFIER*. L'apprentissage supervisé est une technique où l'on cherche à produire automatiquement des règles à partir d'un jeu de données d'apprentissage contenant des exemples de cas déjà traités. Chaque méthode constitue un fichier **.tdm** qui sera exécuté en batch mode par le logiciel *TANAGRA*.

Afin d'évaluer les performances du modèle de classification, et donc du fichier d'apprentissage, des tests de bonne reconnaissance sont réalisés sur des fichiers contenant des objets dont l'identification est connue (fonction *Evaluation*, Fig. 6).

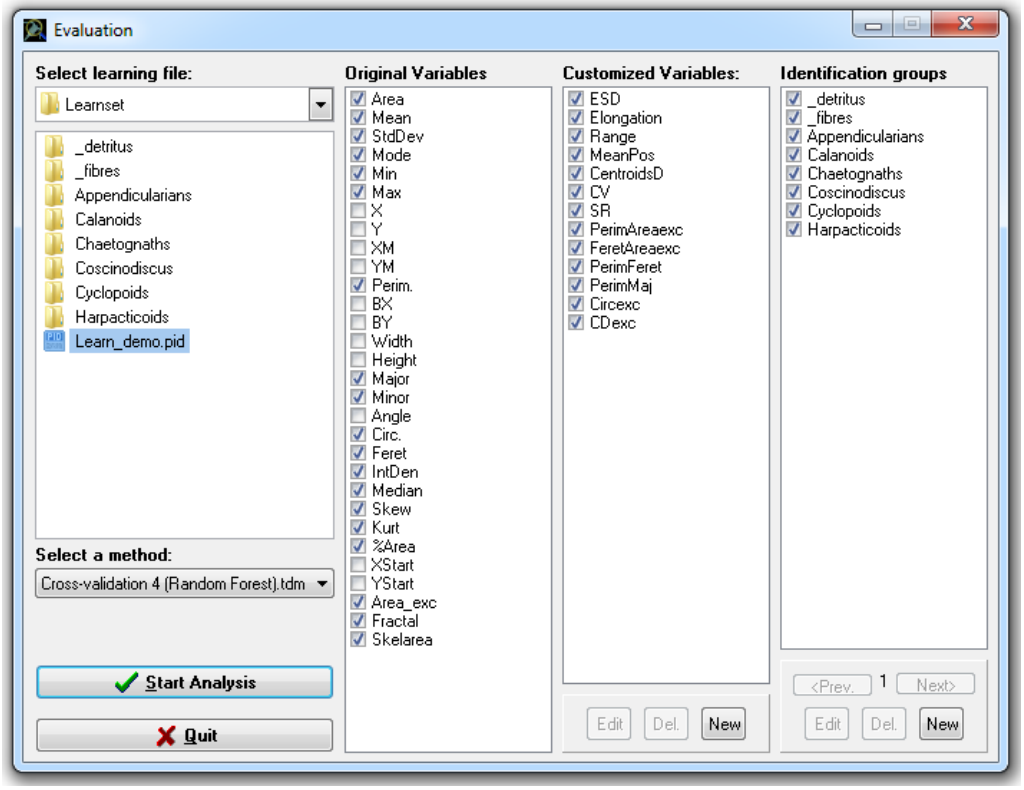


Fig. 6 : Fenêtre Evaluation

La fonction *Prediction* (Fig. 7), permet d'identifier de façon automatique les objets d'un ou plusieurs échantillons, à partir du fichier d'apprentissage (Learn_xxxx.pid) et de la méthode d'apprentissage (.tdm) que l'on aura sélectionné.

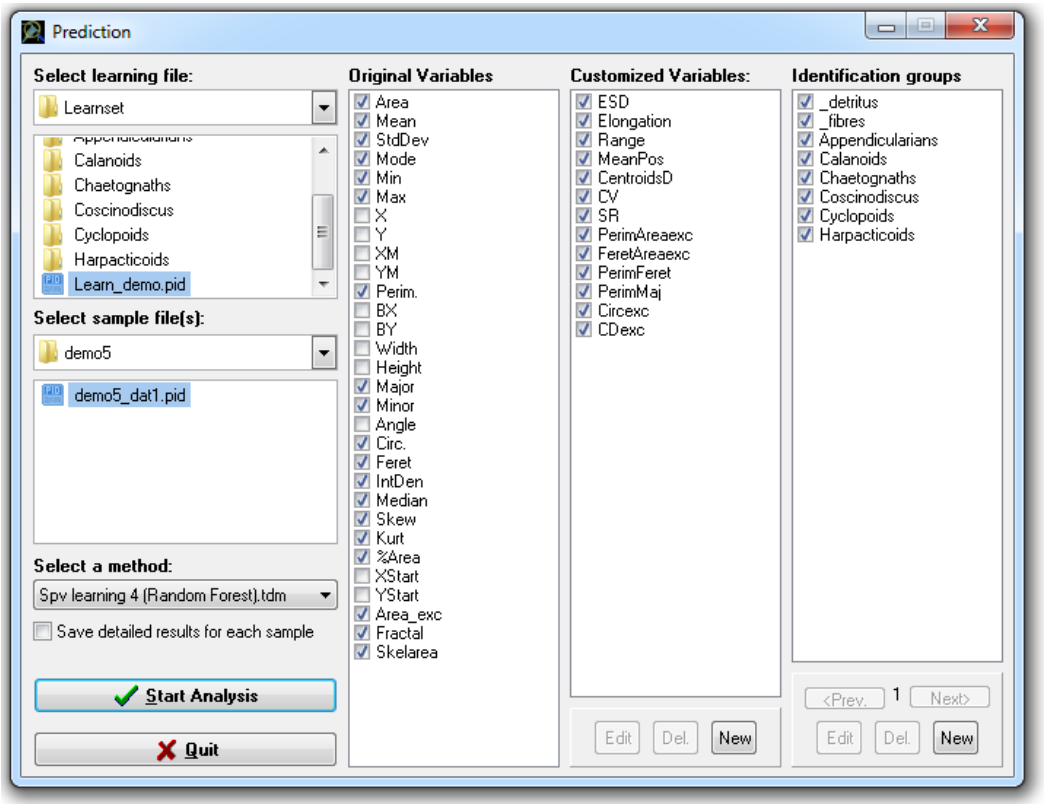


Fig. 7 : Fenêtre Prédiction

3.3 *La fonction* Validation

Une fois l'identification automatique des objets d'un échantillon terminée, il est nécessaire de visualiser et corriger les résultats de la prédiction manuellement. Cette fonction *Validation* se déroule en deux étapes :

- Une première fenêtre permettant de voir le rapport d'analyse de la prédiction et de sélectionner les éléments nécessaires pour trier les vignettes d'un échantillon en fonction de la prédiction (Fig. 8) ;

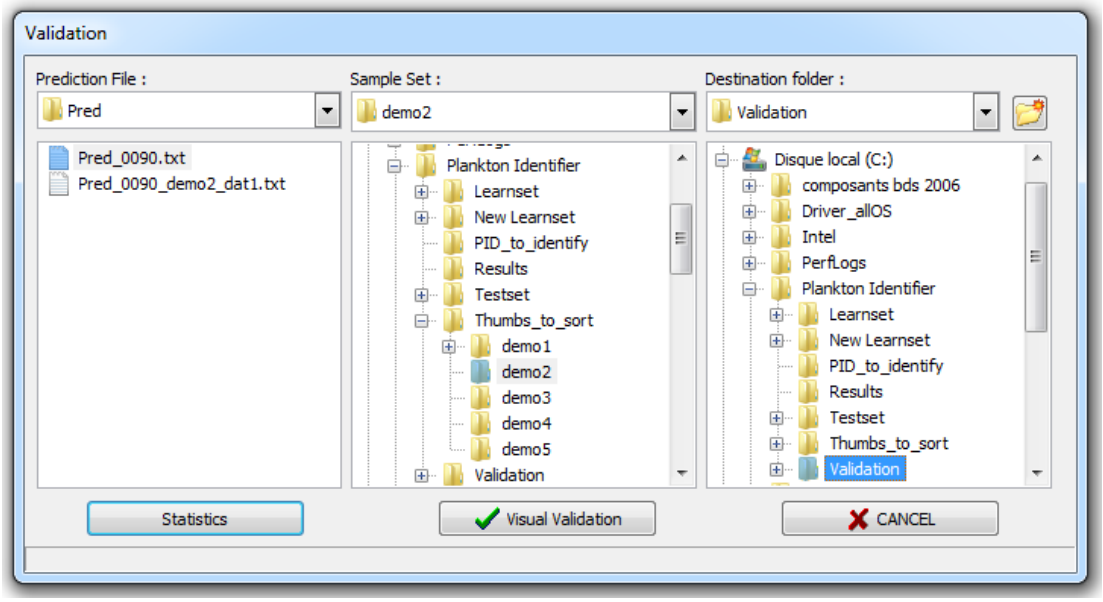


Fig. 8 : Première fenêtre de la fonction Validation

- Une deuxième fenêtre permettant la correction/validation des vignettes qui ont été automatiquement classées dans le dossier « Predicted set » en les déplaçant vers le dossier « Validated set » (Fig. 9) ;

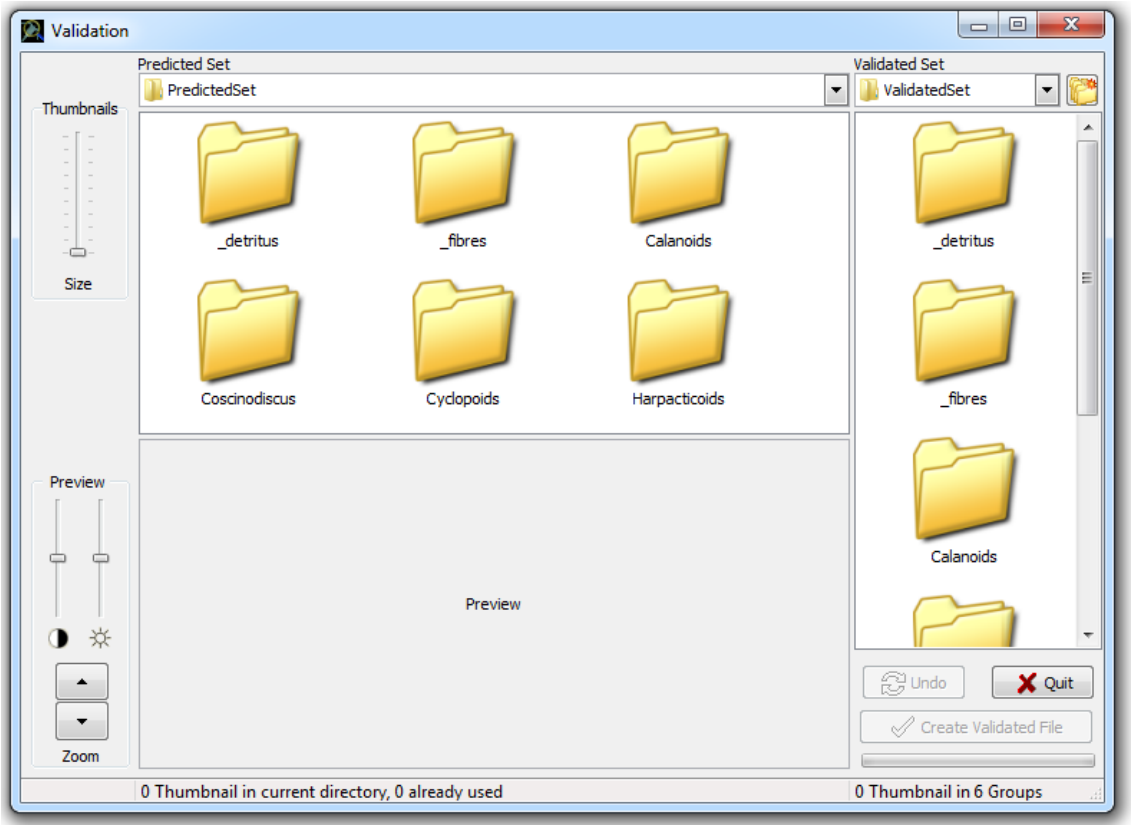


Fig. 9 : Deuxième fenêtre de la fonction Validation

4. Le logiciel *PID VIEWER*

Le logiciel *PID VIEWER* a également été conçu en *Delphi* par le Dr. Stéphane Gasparini. Il permet de lire les fichiers .pid (cf. Annexe II & III) de manière conviviale, en distinguant les informations relatives à l'échantillon et aux traitements qui lui sont appliqués et qui se situent en en-tête du fichier (onglet *Header* Fig. 10A), de la partie *Data* qui contient les paramètres morphométriques des objets de l'échantillon (Fig. 10B). Un troisième onglet permet de visualiser les statistiques de base calculées par paramètres sur l'ensemble des objets de l'échantillon (onglet *Summary*, Fig. 10C). *PID VIEWER* n'est pas un éditeur de texte, l'édition manuelle d'un fichier .pid n'est pas recommandée car il y a un risque de le corrompre, mais si nécessaire, les personnes peuvent utiliser n'importe quel éditeur de texte habituel.

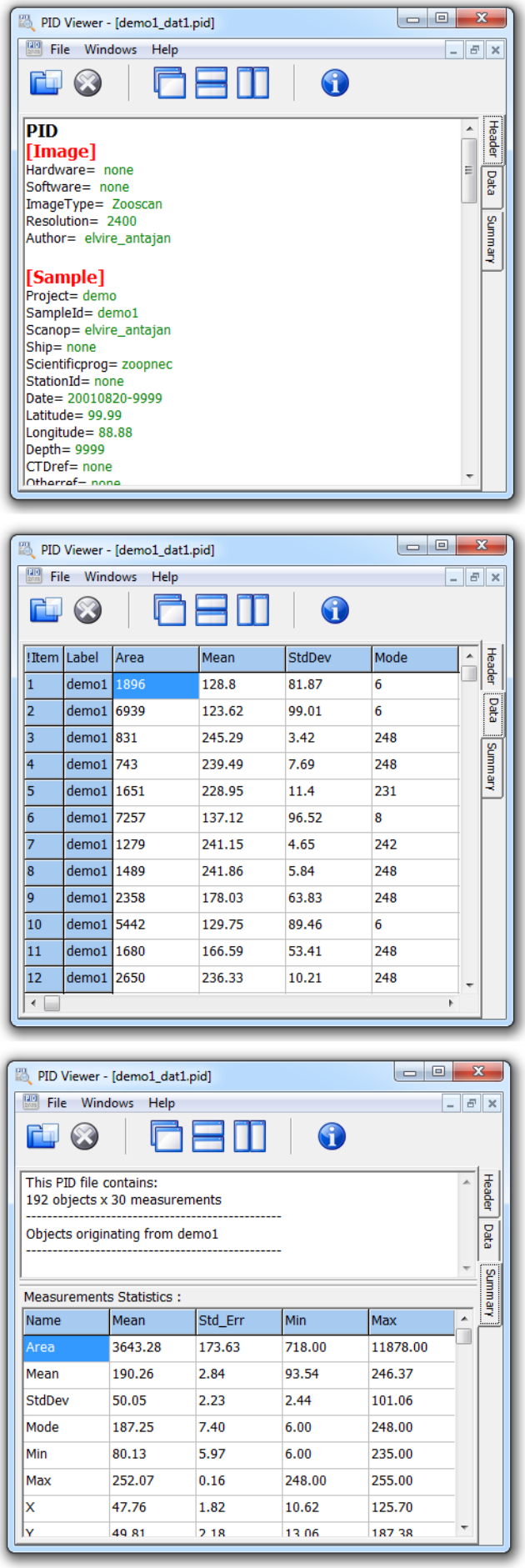


Fig. 10 : Les 3 onglets de lecture d'un fichier .pid : A) Header, B) Data, C) Summary

III. Définition des besoins

Les besoins ont été définis avec les maîtres d'ouvrage Elvire Antajan, Franck Coppin et Stéphane Gasparini.

L'objectif principal du stage était de mettre à jour le logiciel ***PLANKTON IDENTIFIER v1.3.0*** en apportant des améliorations de performances, des corrections, et de nouvelles fonctionnalités. L'interface était aussi à améliorer afin qu'elle soit plus simple et plus intuitive. Les besoins cités ci-dessous ont été classés par ordre de priorité.

1. Les besoins indispensables

1.1 Résoudre le problème de gestion de mémoire

Ce besoin était le plus urgent et devait donc être résolu en première priorité. Les personnes travaillant dans un système win32 sous *Windows XP* ou *Windows 2000* ne pouvaient pas travailler avec la version 1.3.0 de *PLANKTON IDENTIFIER*, un message d'erreur « Ressource système insuffisant » apparaissait lors des manipulations des vignettes.

1.2 Modification de la fonction Validation

1.2.1 Fenêtre de sélection (cf. Fig. 8)

- Renommer le bouton « Statistics » en « View analysis Report » ;
- Le bouton « Visual validation » doit simplement trier les vignettes de l'échantillon dans le dossier de destination choisi par l'utilisateur, sans créer de dossier intitulé par défaut « Predicted set », ni créer un dossier « Validated set » contenant des sous dossier vides ;
- Créer sous le bouton « Visual validation » une checkbox « Keep unsorted thumbnails » permettant de garder les vignettes non triées ou au contraire de les supprimer. Les maîtres d'ouvrage ont précisé qu'il faut seulement supprimer les images sous format JPEG qui ont été copiées dans le « Destination folder ».

1.2.2 Fenêtre de création d'une validation (cf. Fig. 9)

- Comme le dossier « Validated set » n'est plus utile, la fenêtre *Validation* ne doit garder que la partie de droite (qu'il faudra renommer « Predicted set ») et la partie permettant de visualiser la vignette sélectionnée. La validation se faisant désormais grâce à l'option « move to » par clic droit de la souris sur la vignette qui n'a pas été classée dans le bon dossier (comme dans l'option *Learning* quand on corrige l'ensemble d'apprentissage) ;
- La fonction « delete » disponible depuis le clic droit de la souris doit être activé lorsque on sélectionne un dossier ne contenant pas de fichier JPEG (mais qui peut contenir un fichier _dat1.pid), et désactiver lorsque on sélectionne une vignette ou un dossier contenant des vignettes ;
- Faire en sorte que l'image affichée en grand lors d'une sélection multiple soit toujours celle de la dernière sélection (le code affichait la première de la liste ce qui obligeait à commencer par la fin).

1.2.3 Modification d'une validation

L'usage montre que l'opérateur doit bien souvent revenir sur une validation déjà effectuée (soit pour faire un second contrôle par un autre expert, soit pour faire une identification plus poussée d'un groupe en particulier, etc.). Il faut donc que l'on puisse avoir accès à la deuxième fenêtre de validation directement depuis la fenêtre principale de *PLANKTON IDENTIFIER*.

- Lorsque nous cliquons sur *Validation* dans la fenêtre principale de *PLANKTON IDENTIFIER*, il faut faire apparaître un message demandant (Fig. 11) :

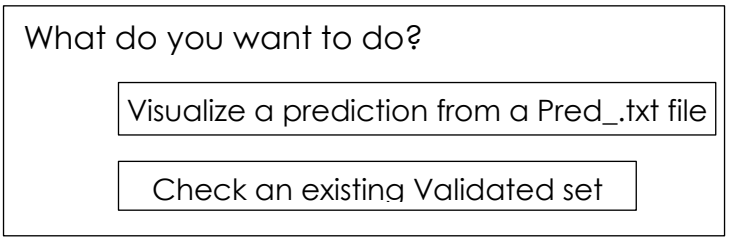


Fig. 11 : Maquette proposant deux possibilités de réponses

- Si la première option est sélectionnée, l'utilisateur est redirigé vers la première fenêtre *Validation* qui a été modifiée selon les indications du point 1.2.1. Si la

deuxième option est sélectionnée, l'utilisateur est redirigé vers une nouvelle fenêtre *Validation (Fig. 12)* :

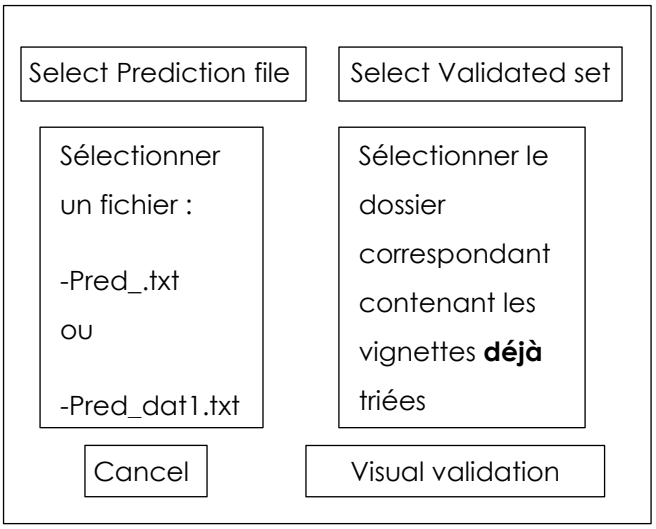


Fig. 12 : Maquette d'une nouvelle fenêtre de sélection

Lorsque l'on clique sur « Visual validation », cela ouvre le dossier de validation choisi dans la deuxième fenêtre de *Validation*.

1.3 Modification de la fonction Learning

- Créer un bouton « Dossier parent » dans le volet *Sample set* (partie de gauche cf. Fig. 5) et dans le volet *Learning set* (partie de droite, cf. Fig. 5), car à l'usage il est assez fastidieux de devoir passer par le dérouleur pour revenir au dossier parent. Ce bouton pourrait être placé à droite du dérouleur (comme c'est déjà le cas pour le bouton « Créer un nouveau dossier » dans le volet *Learning set* ;
- Faire que le curseur *Thumbnails* à gauche pilote à la fois la taille des vignettes ou des dossiers dans les volets *Sample set* et *Learning set*. Faire que la taille minimale soit plus petite ;
- Corriger le bug du focus du dossier de destination lors de son survol pour un glisser-déposer (le dossier ne prend pas toujours le focus selon le trajet de la souris, d'où échec si l'on dépose le document) ;
- Faire en sorte que la vignette qui s'affiche lors d'une sélection multiple soit toujours la dernière qui a été sélectionnée (actuellement le code affiche la première de la liste ce qui oblige à commencer par la fin).

1.4 Modification des fonctions Evaluation *et* Prediction

- Faire qu'à la fin de l'analyse une fenêtre s'ouvre automatiquement pour permettre de visualiser directement le rapport d'analyse html produit par le logiciel *TANAGRA* ;
- Pour les options *Prediction* et *Evaluation* : Créer un bouton « Edit tdm File » pour éditer les fichiers .tdm et pouvoir les modifier si besoin ;
- Supprimer les checkbox dans le volet *Identification group*, car tous les groupes de la liste doivent toujours être utilisés ;
- Renommer le bouton « Start Analysis » en « Run »
- Pour l'option *Evaluation* uniquement : Le fichier d'évaluation créé doit commencer par Eval_.txt et non par Analysis_.txt (à modifier dans la fenêtre enregistrer-sous qui s'ouvre lorsque on clique sur « Run ») ;
- Pour l'option *Prediction* uniquement : Le fichier de prédiction créé doit commencer par Pred_.txt et non par Analysis_.txt (à modifier dans la fenêtre enregistrer-sous qui s'ouvre lorsqu'on clique sur « Run »).

2. Les besoins optionnels

2.1 Amélioration à apporter au logiciel PID VIEWER

Le logiciel permet de lire et comparer les fichiers .pid de façon conviviale, néanmoins à l'usage on s'est rendu compte qu'il fallait parfois compléter/modifier certaines informations contenues dans la partie *Header* des fichiers.

- Rendre éditable l'onglet *Header* du logiciel, mais avec la création d'un fichier .old pour la traçabilité.

2.2 Ajouter une nouvelle option Compilation

Cette nouvelle fonction qui sera accessible dans la fenêtre principale de *PLANKTON IDENTIFIER* (à la suite du bouton Validation) permettra d'obtenir un fichier .txt contenant les informations compilées de un ou plusieurs fichiers validés. L'aspect général de la fenêtre correspondante avait été défini sur papier.

IV. Environnement de travail

1. Gestion de projet

Afin de mener à terme le projet et rendre un travail de qualité, la méthode AGILE qui autorise un processus itératif a été appliquée. Elle vise en priorité la satisfaction réelle du client aux termes d'un contrat de développement.

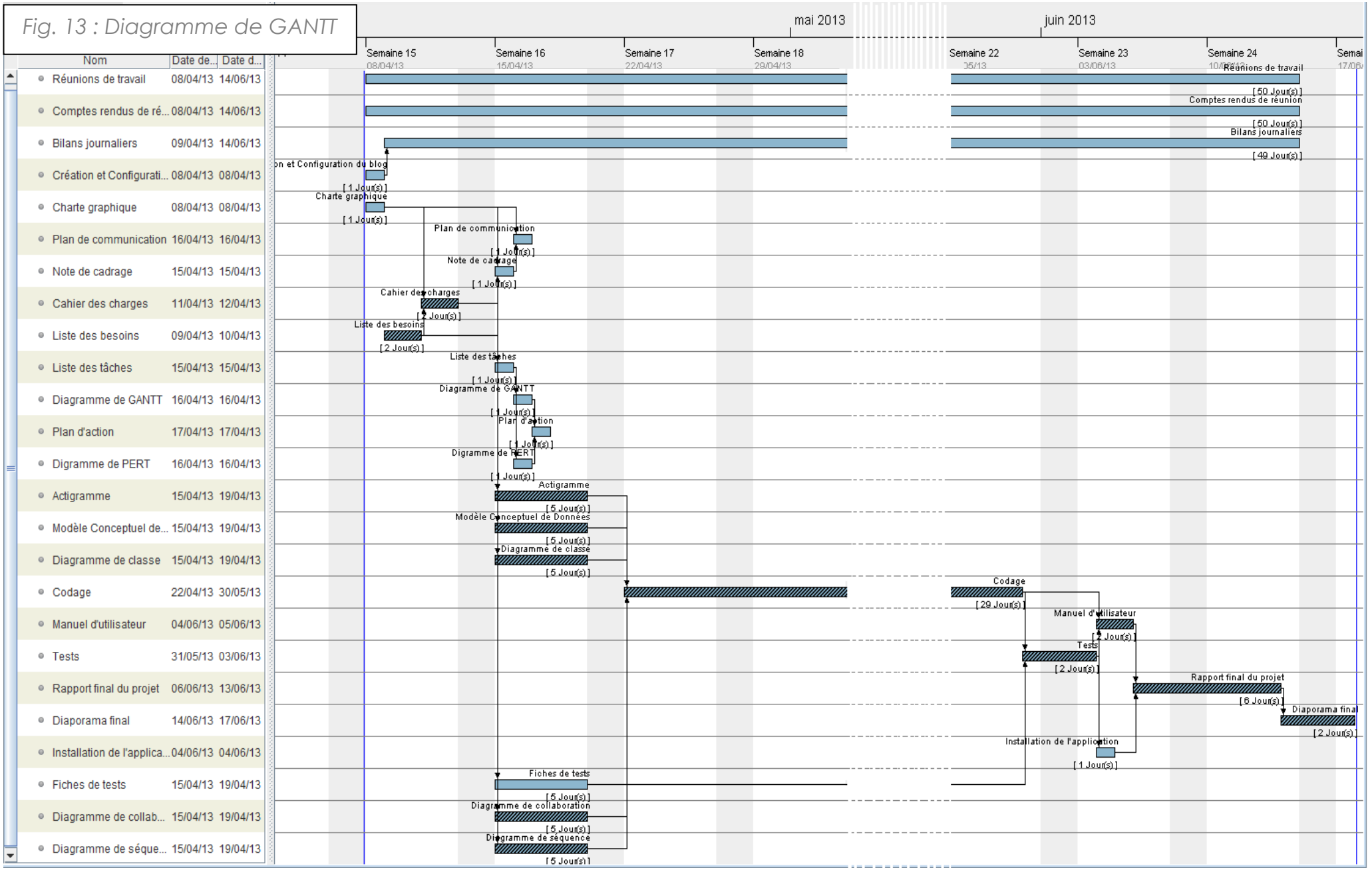
1.1 Planning prévisionnel

Le planning prévisionnel présenté en figure 13 planifie cinq grandes tâches :

- Liste les besoins ;
- Rédaction des livrables ;
- Analyse de l'existant ;
- Développement ;
- Rédaction du rapport et préparation de la soutenance.

Le chemin critique, qui est grisé dans le planning prévisionnel, est un chemin sur lequel aucune tâche ne doit avoir de retard pour ne pas retarder l'ensemble du projet.

Fig. 13 : Diagramme de GANTT



1.2 Journal de bord

Un blog a été mis en place pour suivre au mieux l'évolution du projet. Ce blog est disponible à l'adresse suivante : **http://stagepkid.wordpress.com**. Y sont stockés les bilans de séances, les documents livrables, les analyses et les comptes rendus de réunions.

Ce blog est à accès libre et n'est pas indexé dans les moteurs de recherche. Toutes les personnes voulant se rendre compte de l'état d'avancement du projet ont été invitées à visiter ces pages. Les cibles potentielles de ce blog pour leur permettre de suivre l'état d'avancement ont été Elvire Antajan, Bénédicte Talon, Franck Coppin et Stéphane Gasparini.

2. Environnement technique

2.1 Méthodes et Langage de modélisation utilisés

Afin de procéder à l'analyse de l'existant, plusieurs méthodes ont été appliquées :

- La méthode MERISE, est une méthode d'analyse, de conception et de réalisation de systèmes d'informations informatisés. Elle a été utilisée pour concevoir le modèle conceptuel de données et le modèle opérationnel de traitements ;
- La méthode SADT (*Structured Analysis and Design Technique*), est une méthode d'analyse qui décrit/modélise le cheminement du général vers le particulier et le détaillé d'un système complexe ou d'un processus opératoire. Elle a été utilisée pour réaliser l'actigramme du processus d'identification d'un échantillon présenté en Annexe 1.

De plus, le langage UML (*Unified Modeling Language*) a également été utilisé pour réaliser des diagrammes de séquences.

2.2 Ressources

Un certain nombre de ressources matérielles et immatérielles a été nécessaire pour réaliser le projet :

- Un ordinateur ;
- La suite Microsoft Office.

2.3 Environnement de développement

Le langage de programmation utilisé pour mettre à jour le logiciel *PLANKTON IDENTIFIER* est le langage orienté objet *Delphi.* Ce langage implémente une version orientée objet du langage *Pascal ;*

Le choix de l'IDE (*Integrated Development Environment*) *Delphi* a été une problématique au début du stage (cf. Annexe compte-rendu n°3), mais notre décision c'est portée sur l'IDE *Borland Developer Studio 2006.*

L'apprentissage du langage *Delphi* c'est fait en consultant les sites suivants :

- http://delphi.developpez.com/ ;
- http://www.delphibasics.co.uk/ ;
- http://www.phidels.com/php/index.php3.

En plus des composants standards de l'IDE, la version précédente de *PLANKTON IDENTIFIER* utilisait des composants additionnels, il en sera de même pour la nouvelle version. Les composants utilisent des propriétés pour spécifier le gestionnaire d'événement qui sera exécuté si l'événement se produit. Par convention, le nom d'une propriété d'événement commence par "On", et la propriété est implémentée par un champ au lieu des méthodes de lecture/écriture.

V. Analyse de l'existant

Le logiciel à améliorer ayant été développé par Stéphane Gasparini, il a été nécessaire de consacrer du temps à bien comprendre l'organisation du code et de la gestion des données.

1. Le modèles conceptuel de données

Le Modèle conceptuel de données (MCD), présenté en figure 14, établi une représentation claire des données d'un échantillon et définit les dépendances fonctionnelles de ces données entre elles.

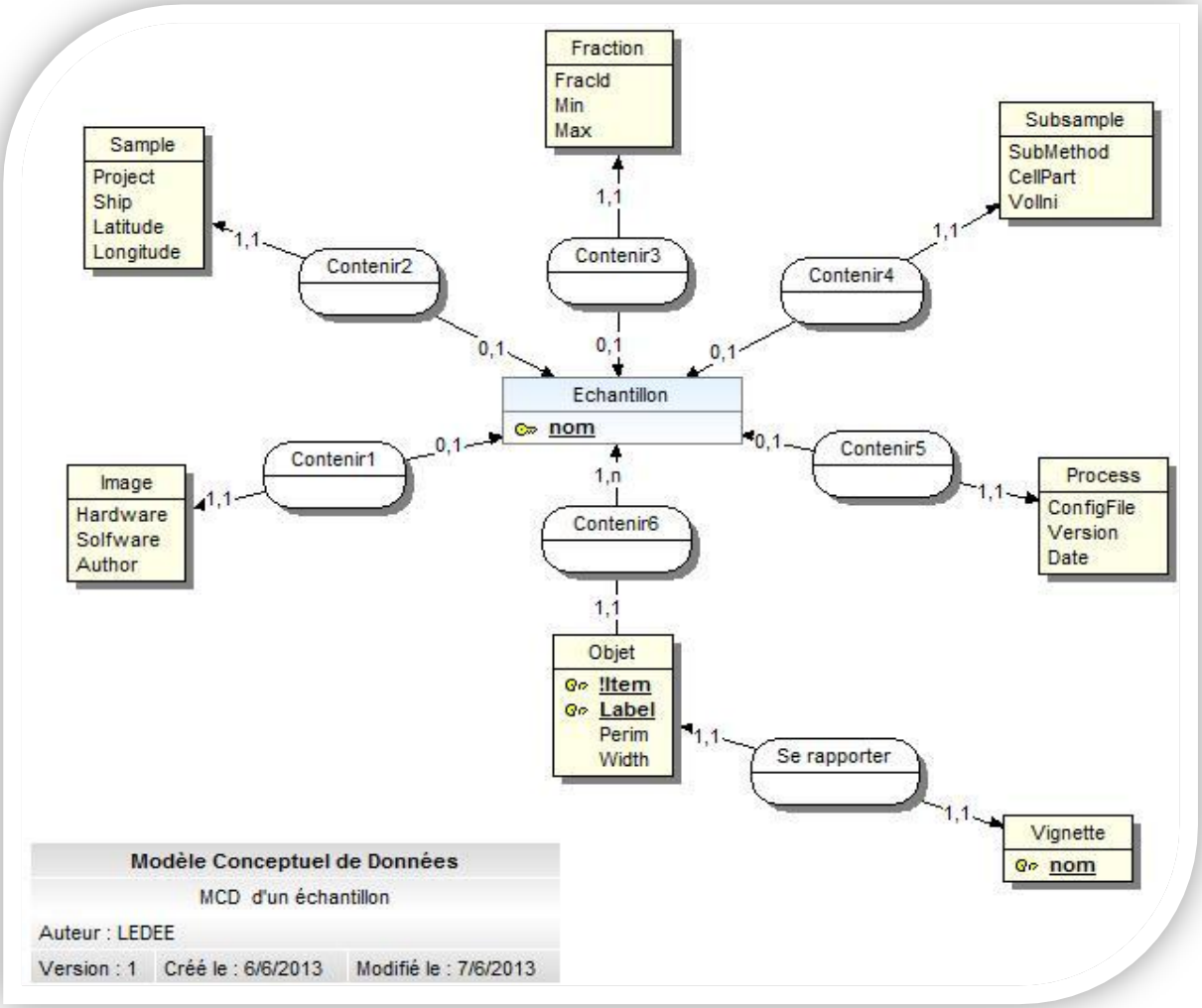
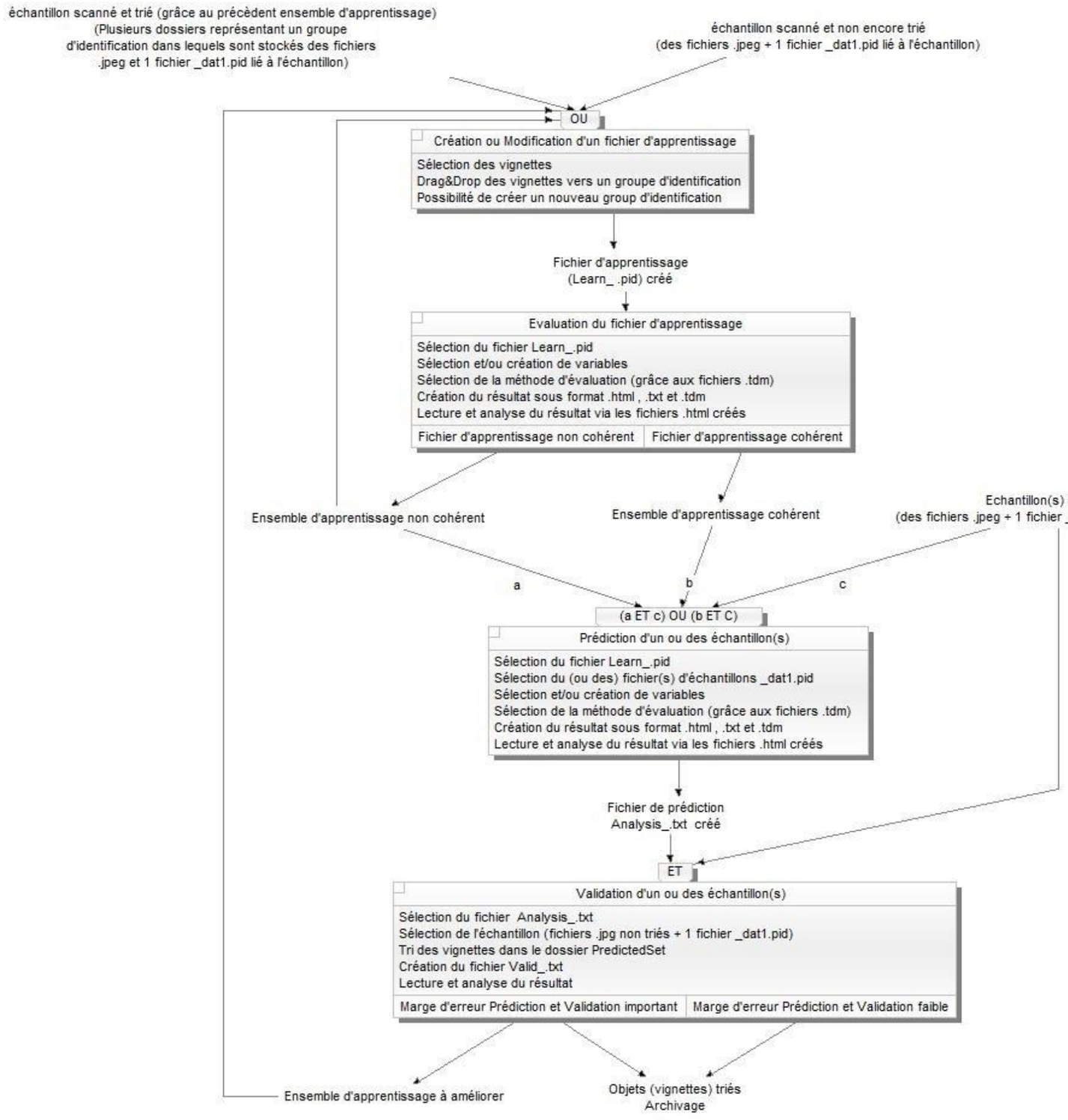


Fig. 14 : Modèle conceptuel de données d'un échantillon

L'échantillon à analyser se distingue de l'échantillon d'apprentissage (*Learning*) par la présence du *Header* dans le fichier PID. Ce qui justifie les cardinalités 0-1 dans le modèle conceptuel ci-dessus.

2. Le modèle opérationnel de traitements

Le Modèle opérationnel de traitements (MOT) présenté ci-dessous (Fig. 15) décrit les principales fonctionnalités assurées par le logiciel *PLANKTON IDENTIFIER*.


échantillon scanné et trié (grâce au précèdent ensemble d'apprentissage)
(Plusieurs dossiers représentant un groupe d'identification dans lequels sont stockés des fichiers .jpeg et 1 fichier _dat1.pid lié à l'échantillon)
échantillon scanné et non encore trié
(des fichiers .jpeg + 1 fichier _dat1.pid lié à l'échantillon)
OU
Création ou Modification d'un fichier d'apprentissage
Sélection des vignettes
Drag&Drop des vignettes vers un groupe d'identification
Possibilité de créer un nouveau group d'identification
Fichier d'apprentissage
(Learn_ .pid) créé
Evaluation du fichier d'apprentissage
Sélection du fichier Learn_.pid
Sélection et/ou création de variables
Sélection de la méthode d'évaluation (grâce aux fichiers .tdm)
Création du résultat sous format .html , .txt et .tdm
Lecture et analyse du résultat via les fichiers .html créés
Fichier d'apprentissage non cohérent
Fichier d'apprentissage cohérent
Ensemble d'apprentissage non cohérent
Ensemble d'apprentissage cohérent
Echantillon(s) à analyser
(des fichiers .jpeg + 1 fichier _dat1.pid lié à l'échantillon)
a
b
c
(a ET c) OU (b ET C)
Prédiction d'un ou des échantillon(s)
Sélection du fichier Learn_.pid
Sélection du (ou des) fichier(s) d'échantillons _dat1.pid
Sélection et/ou création de variables
Sélection de la méthode d'évaluation (grâce aux fichiers .tdm)
Création du résultat sous format .html , .txt et .tdm
Lecture et analyse du résultat via les fichiers .html créés
Fichier de prédiction
Analysis_.txt créé
ET
Validation d'un ou des échantillon(s)
Sélection du fichier Analysis_.txt
Sélection de l'échantillon (fichiers .jpg non triés + 1 fichier _dat1.pid)
Tri des vignettes dans le dossier PredictedSet
Création du fichier Valid_.txt
Lecture et analyse du résultat
Marge d'erreur Prédiction et Validation important
Marge d'erreur Prédiction et Validation faible
Ensemble d'apprentissage à améliorer
Objets (vignettes) triés
Archivage

Modèle opérationnel de traitements (MOT) - AVANT MAJ		
Logiciel Plankton Identifier : Analyse d'un ou des échantillons		
Auteur : Maxime LEDEE		
Version : 1.5	Créé le : 17/4/2013	Modifié le : 26/5/2013

Constitution et évaluation d'un ensemble d'apprentissage ("Learning set")

Analyse d'un échantillon

Fig. 15 : Modèle opérationnel de traitements des fonctionnalités assurées par le logiciel PLANKTON IDENTIFIER (avant modifications)

3. Représentation physique des données

3.1 *Un échantillon*

La représentation physique des données d'un échantillon (Fig. 16) présente l'organisation des données d'un échantillon à la sortie du *ZooScan*. Après avoir scanné l'échantillon via le *ZooScan*, nous obtenons donc plusieurs vignettes sous format JPEG et un seul fichier PID nommé « *nom_echantillon*_**dat1**.pid ». Ce fichier PID possède deux parties : *HEADER* et *DATA* (Annexe II).

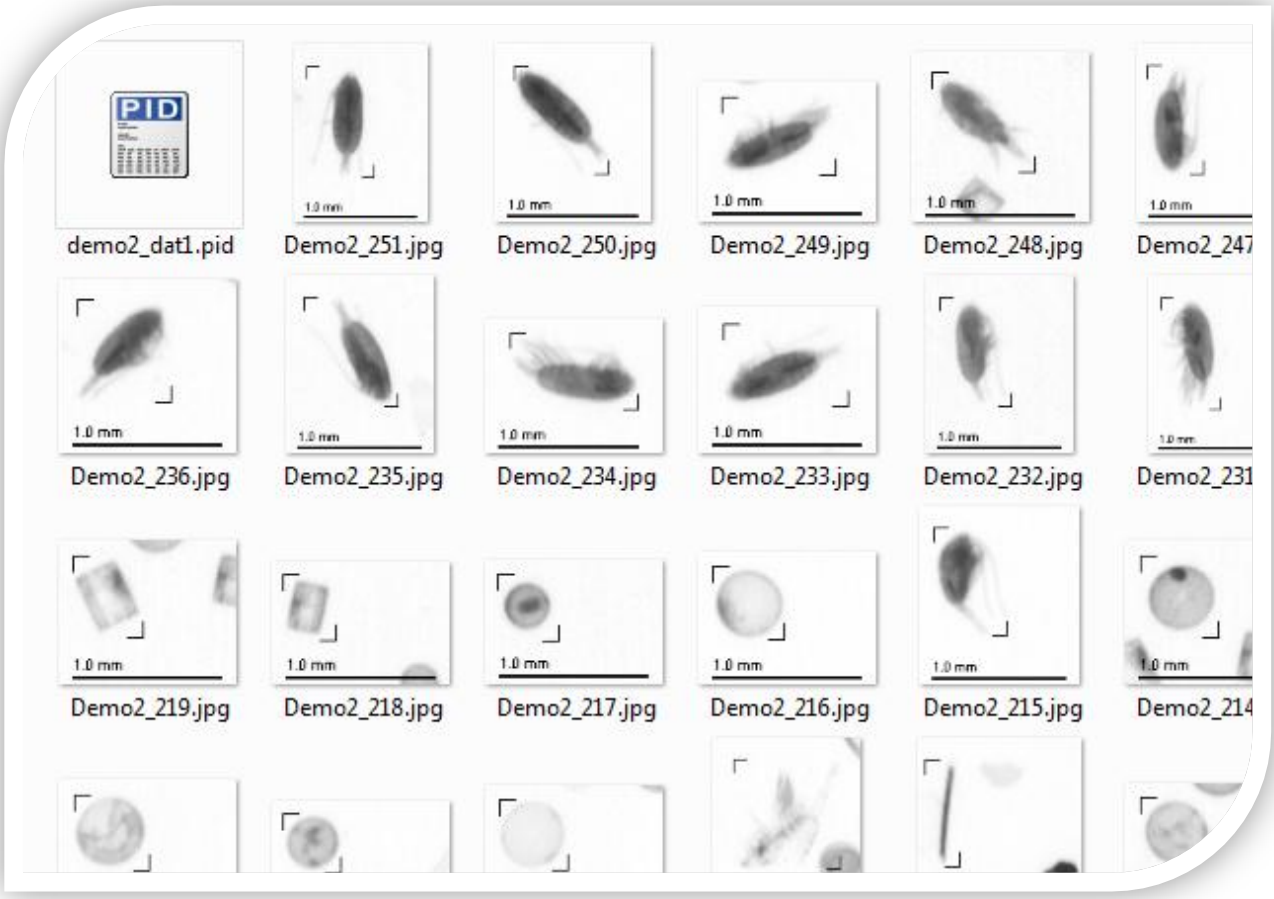


Fig. 16 : Représentation physique des données d'un échantillon

3.2 *Un ensemble d'apprentissage*

La représentation physique des données d'un ensemble d'apprentissage (Fig. 17) présente l'organisation des données d'un jeu d'apprentissage issue de la fonction *Learning* du logiciel *PLANKTON IDENTIFIER*. Un jeu d'apprentissage contient des vignettes triées selon le groupe d'identification (exemples : *Cyclopoids*, _detritus, ...), les fichiers

« *nom_echantillon*_dat1.pid » des échantillons auxquels ces vignettes appartiennent et le fichier d'apprentissage « Learn_xxxx.pid » qui contient uniquement les informations liées aux vignettes présentes dans le jeu d'apprentissage. Contrairement au fichier PID nommé « *nom_echantillon*_**dat1**.pid » qui contient l'ensemble des informations d'un **seul** échantillon, le fichier PID appelé « **Learn**_xxxx.pid » contient, dans la partie DATA, des objets provenant de **plusieurs** échantillons identifiés selon leur groupe taxinomique d'appartenance dans une colonne nommée *Ident* (Annexe III).

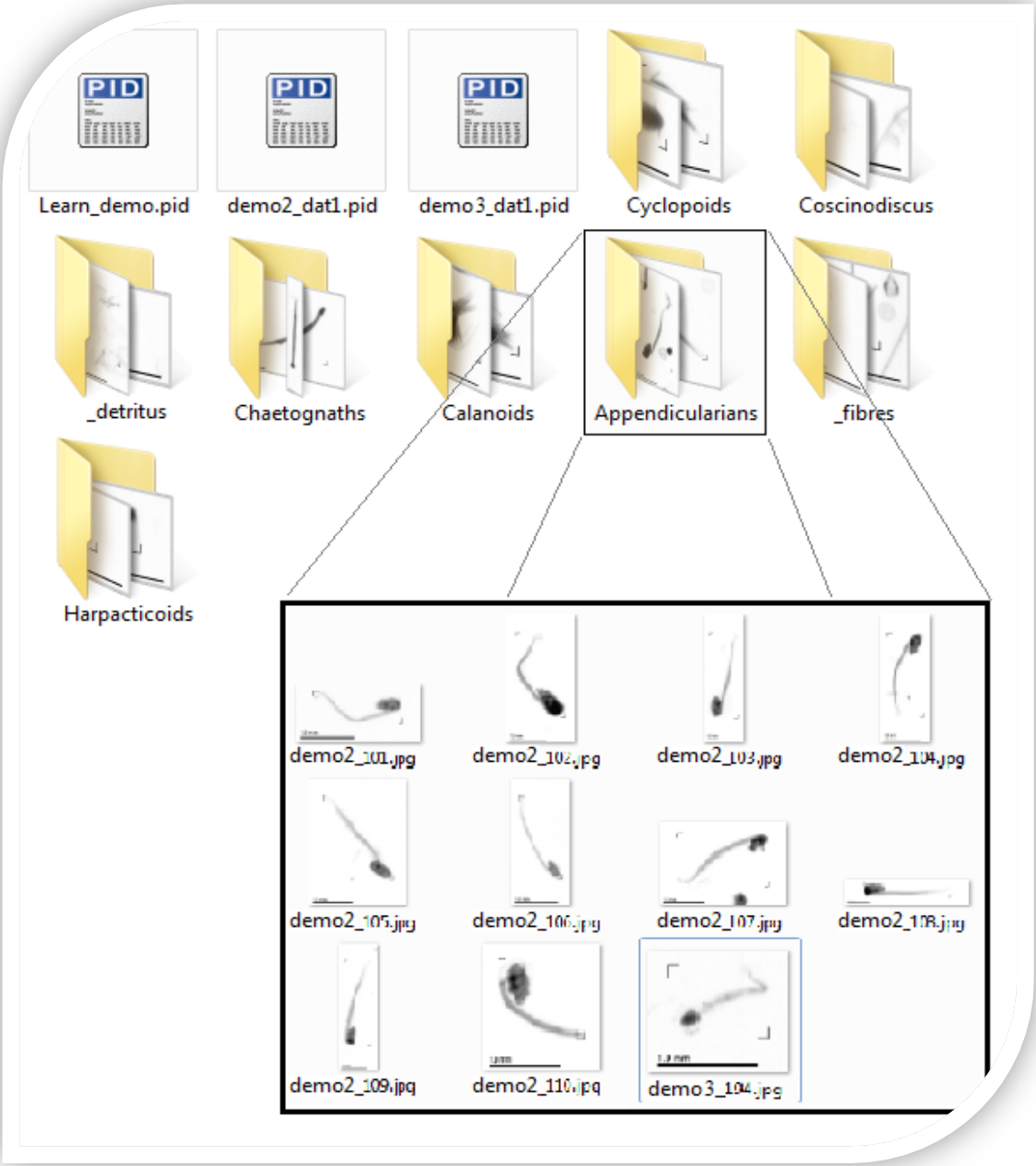


Fig. 17 : Représentation physique des données d'un ensemble d'apprentissage

3.3 Un dossier de résultat d'analyse

La représentation physique des données d'un dossier de résultat d'analyse présente l'organisation des résultats d'analyses issus des fonctions *Evaluation* et *Prediction* de *PLANKTON IDENTIFIER*. Ce dossier comprend un rapport d'analyse html produit par le logiciel *TANAGRA*, un fichier *Analysis_xxxx.txt* qui contient les données qui ont été analysées et un fichier tdm qui contient le calcul d'analyse qui a été effectué sur ces données (Fig. 18).

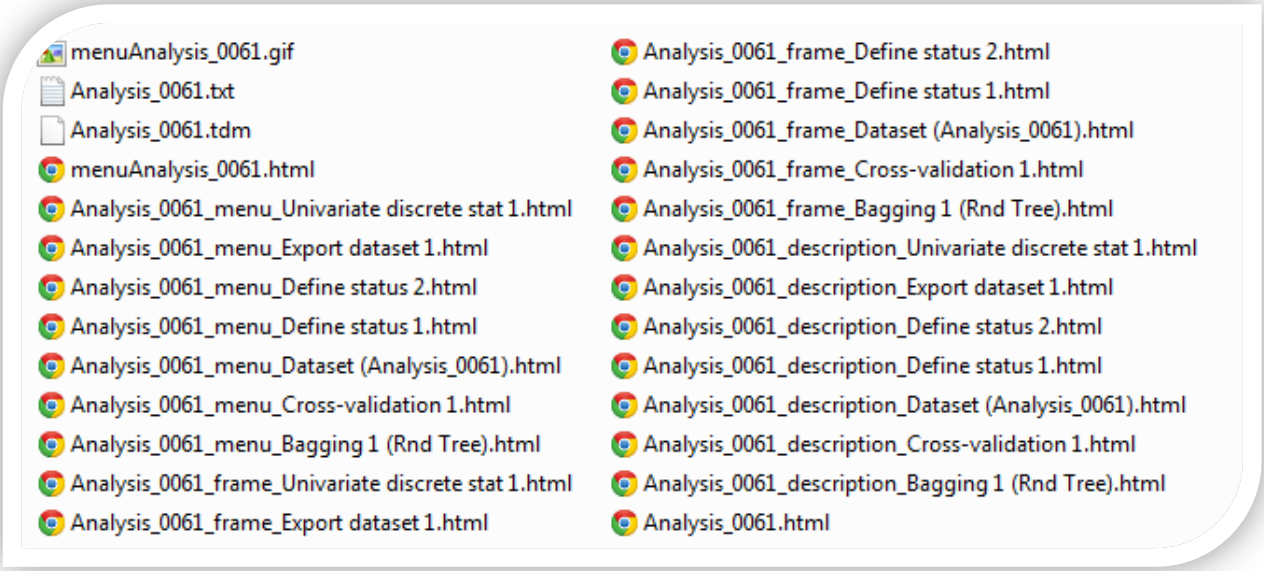


Fig. 18 : Représentation physique des données d'un dossier Evaluation

3.4 Un dossier de validation

La représentation physique des données d'un dossier validation (Fig. 19) présente l'organisation des données issue de la fonction *Validation* de *PLANKTON IDENTIFIER*. Un fichier PID nommé « *nom_echantillon*_**dat1**.pid » est stocké dans chaque groupe d'identification contenant des vignettes appelées « *nom_echantillon*_xxxx.jpg ».

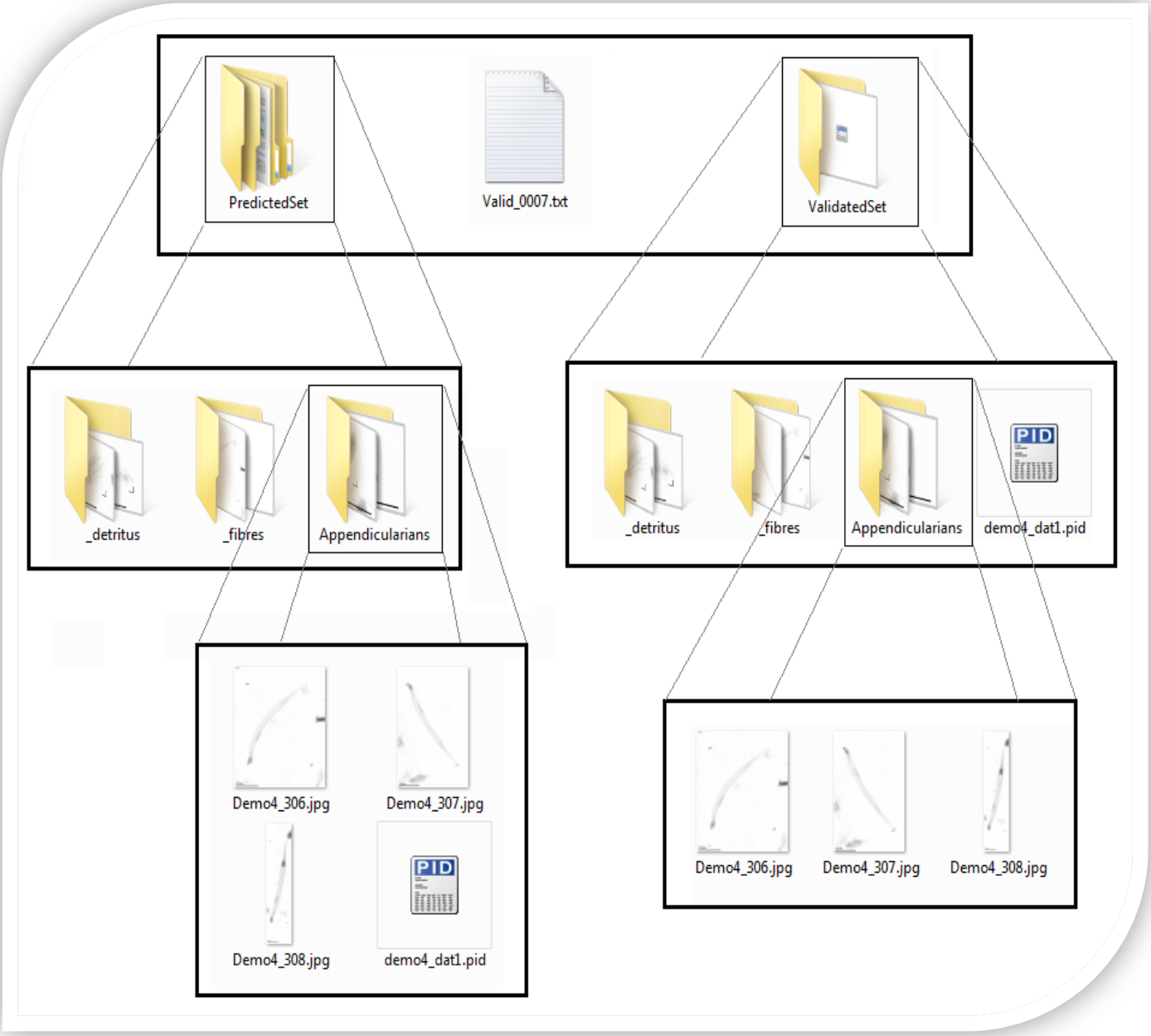


Fig. 19 : Représentation physique des données d'un dossier Validation

4. Les diagrammes de séquences

Les diagrammes de séquences présentés ci-dessous montrent respectivement une vue dynamique de l'interaction entre l'utilisateur et le logiciel *PLANKTON IDENTIFIER* des scénarios de la fonction *Learning (Fig. 20)*, *Evaluation (Fig. 21)*, Prediction *(Fig. 22)*, et Validation *(Fig. 23)*.

Diagramme de séquence

La fonction Learning AVANT MAJ

Auteur : LEDEE

Version : 1.3 | Créé le : 18/4/2013 | Modifié le : 26/5/2013

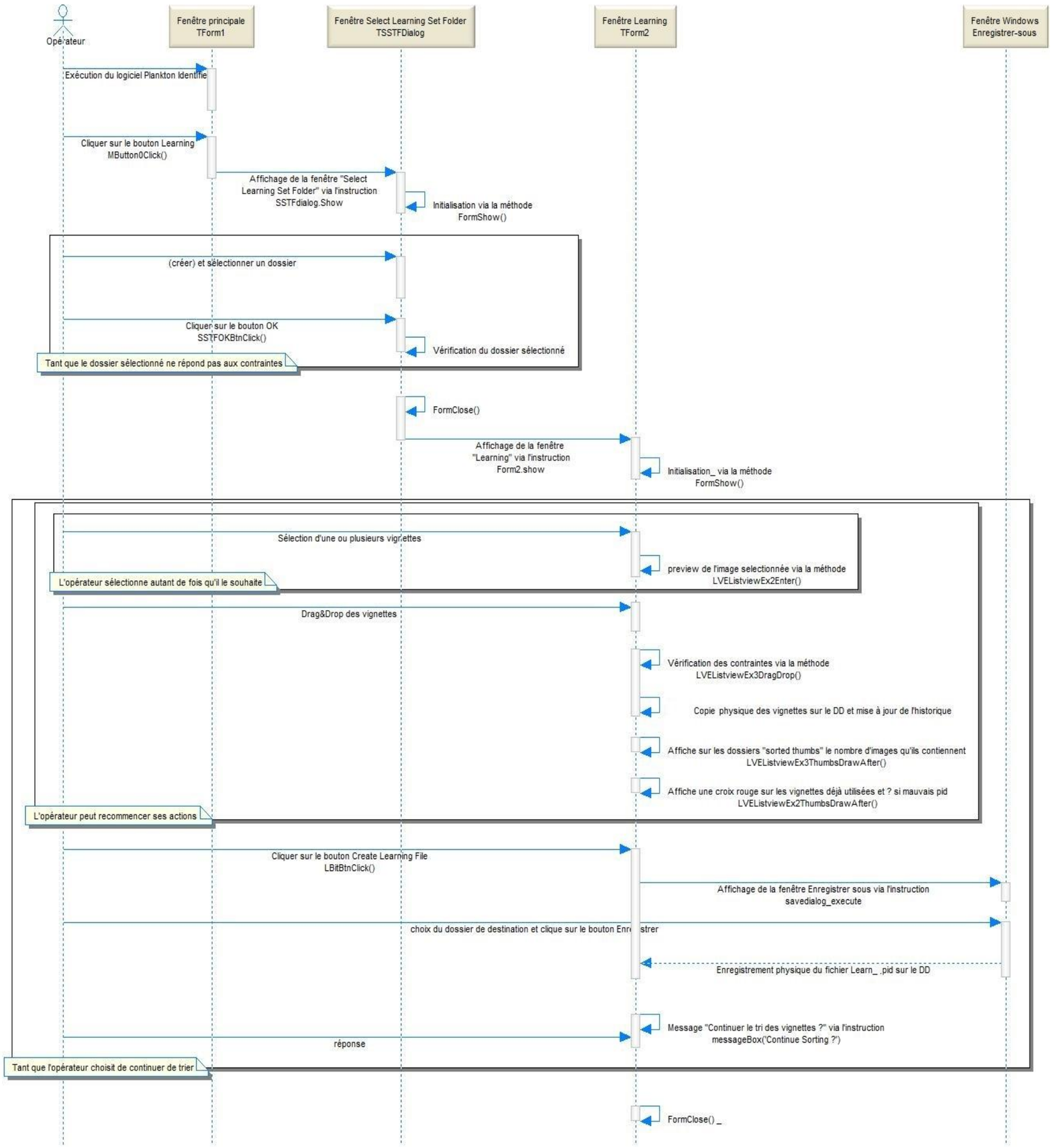

Opérateur
Fenêtre principale TForm1
Fenêtre Select Learning Set Folder TSSTFDialog
Fenêtre Learning TForm2
Fenêtre Windows Enregistrer-sous
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Learning MButton0Click()
Affichage de la fenêtre "Select Learning Set Folder" via l'instruction SSTFdialog.Show
Initialisation via la méthode FormShow()
(créer) et sélectionner un dossier
Cliquer sur le bouton OK SSTFOKBtnClick()
Vérification du dossier sélectionné
Tant que le dossier sélectionné ne répond pas aux contraintes
FormClose()
Affichage de la fenêtre "Learning" via l'instruction Form2.show
Initialisation_ via la méthode FormShow()
Sélection d'une ou plusieurs vignettes
preview de l'image selectionnée via la méthode LVEListviewEx2Enter()
L'opérateur sélectionne autant de fois qu'il le souhaite
Drag&Drop des vignettes
Vérification des contraintes via la méthode LVEListviewEx3DragDrop()
Copie physique des vignettes sur le DD et mise à jour de l'historique
Affiche sur les dossiers "sorted thumbs" le nombre d'images qu'ils contiennent LVEListviewEx3ThumbsDrawAfter()
Affiche une croix rouge sur les vignettes déjà utilisées et ? si mauvais pid LVEListviewEx2ThumbsDrawAfter()
L'opérateur peut recommencer ses actions
Cliquer sur le bouton Create Learning File LBitBtnClick()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog_execute
choix du dossier de destination et clique sur le bouton Enregistrer
Enregistrement physique du fichier Learn_ .pid sur le DD
Message "Continuer le tri des vignettes ?" via l'instruction messageBox('Continue Sorting ?')
réponse
Tant que l'opérateur choisit de continuer de trier
FormClose() _

Fig. 20 : Diagramme de séquence de la fonction Learning *(avant modifications)*

Diagramme de séquence de l'existant		
La fonction Evaluation		
Auteur : LEDEE		
Version : 1.3	Créé le : 18/4/2013	Modifié le : 26/5/2013

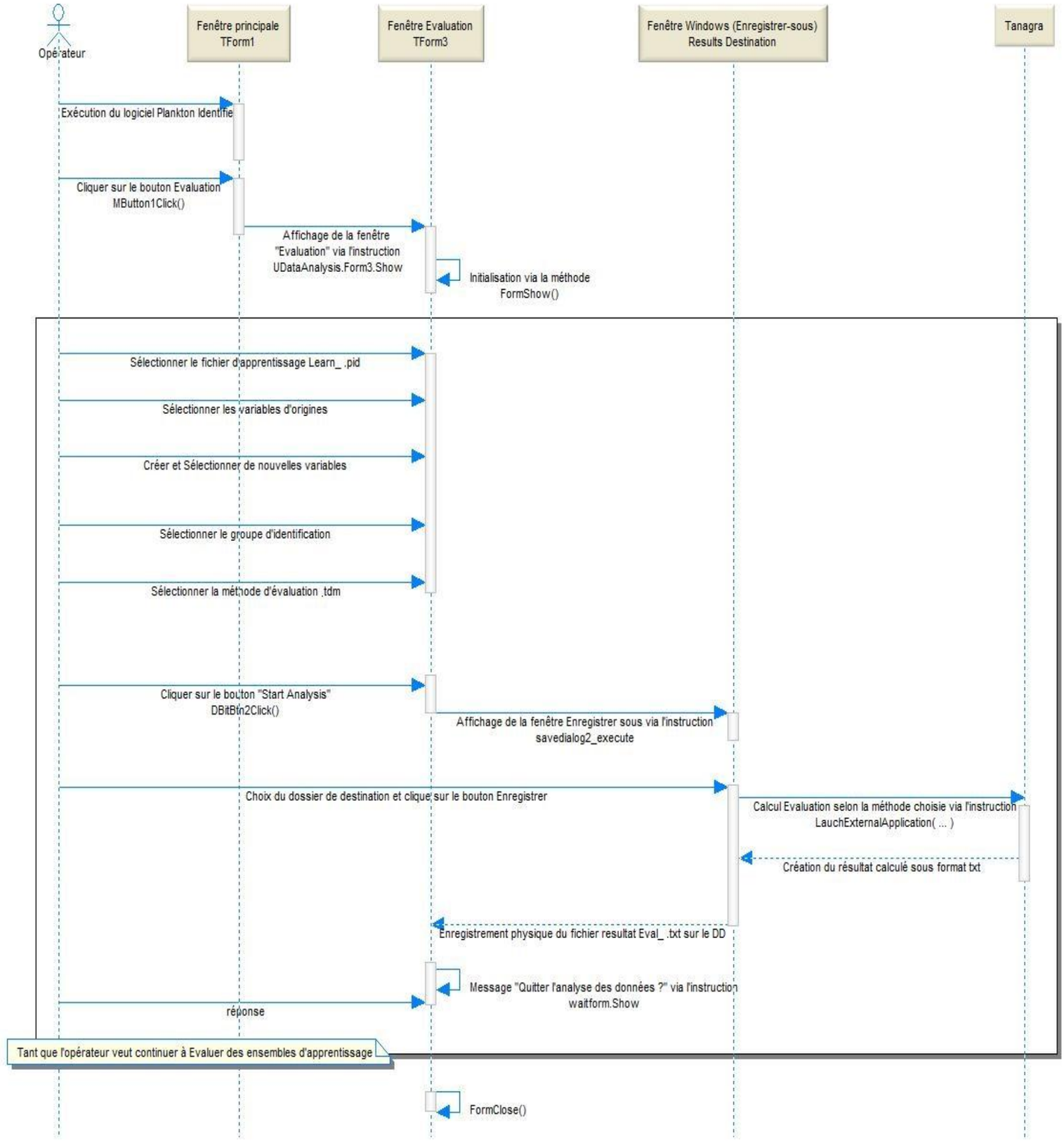

Opérateur
Fenêtre principale TForm1
Fenêtre Evaluation TForm3
Fenêtre Windows (Enregistrer-sous) Results Destination
Tanagra
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Evaluation MButton1Click()
Affichage de la fenêtre "Evaluation" via l'instruction UDataAnalysis.Form3.Show
Initialisation via la méthode FormShow()
Sélectionner le fichier d'apprentissage Learn_ .pid
Sélectionner les variables d'origines
Créer et Sélectionner de nouvelles variables
Sélectionner le groupe d'identification
Sélectionner la méthode d'évaluation .tdm
Cliquer sur le bouton "Start Analysis" DBitBtn2Click()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog2_execute
Choix du dossier de destination et clique sur le bouton Enregistrer
Calcul Evaluation selon la méthode choisie via l'instruction LauchExternalApplication(...)
Création du résultat calculé sous format txt
Enregistrement physique du fichier resultat Eval_ .txt sur le DD
Message "Quitter l'analyse des données ?" via l'instruction waitform.Show
réponse
Tant que l'opérateur veut continuer à Evaluer des ensembles d'apprentissage
FormClose()

Fig. 21 : Diagramme de séquence de la fonction Evaluation *(avant modifications)*

Diagramme de séquence de l'existant		
La fonction Prediction		
Auteur : LEDEE		
Version : 1.3	Créé le : 18/4/2013	Modifié le : 26/5/2013

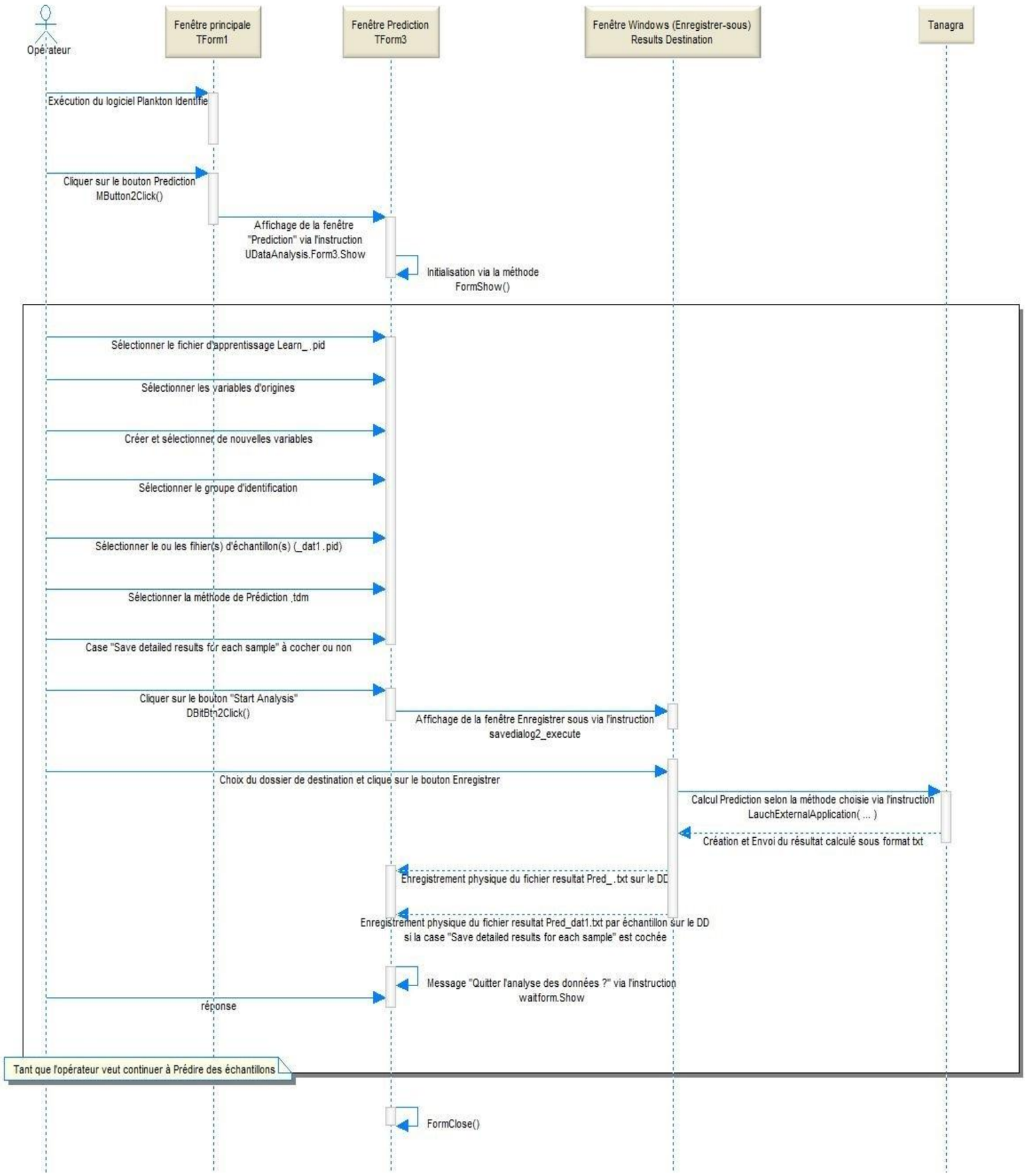

Opérateur
Fenêtre principale TForm1
Fenêtre Prediction TForm3
Fenêtre Windows (Enregistrer-sous) Results Destination
Tanagra
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Prediction MButton2Click()
Affichage de la fenêtre "Prediction" via l'instruction UDataAnalysis.Form3.Show
Initialisation via la méthode FormShow()
Sélectionner le fichier d'apprentissage Learn_.pid
Sélectionner les variables d'origines
Créer et sélectionner de nouvelles variables
Sélectionner le groupe d'identification
Sélectionner le ou les fihier(s) d'échantillon(s) (_dat1.pid)
Sélectionner la méthode de Prédiction .tdm
Case "Save detailed results for each sample" à cocher ou non
Cliquer sur le bouton "Start Analysis" DBitBtn2Click()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog2_execute
Choix du dossier de destination et clique sur le bouton Enregistrer
Calcul Prediction selon la méthode choisie via l'instruction LauchExternalApplication(...)
Création et Envoi du résultat calculé sous format txt
Enregistrement physique du fichier resultat Pred_.txt sur le DD
Enregistrement physique du fichier resultat Pred_dat1.txt par échantillon sur le DD si la case "Save detailed results for each sample" est cochée
Message "Quitter l'analyse des données ?" via l'instruction waitform.Show
réponse
Tant que l'opérateur veut continuer à Prédire des échantillons
FormClose()

Fig. 22 : Diagramme de séquence de la fonction Prediction *(avant modifications)*

Diagramme de séquence

La fonction Validation de l'existant

Auteur : LEDEE

Version : 1.2 | Créé le : 18/4/2013 | Modifié le : 26/5/2013

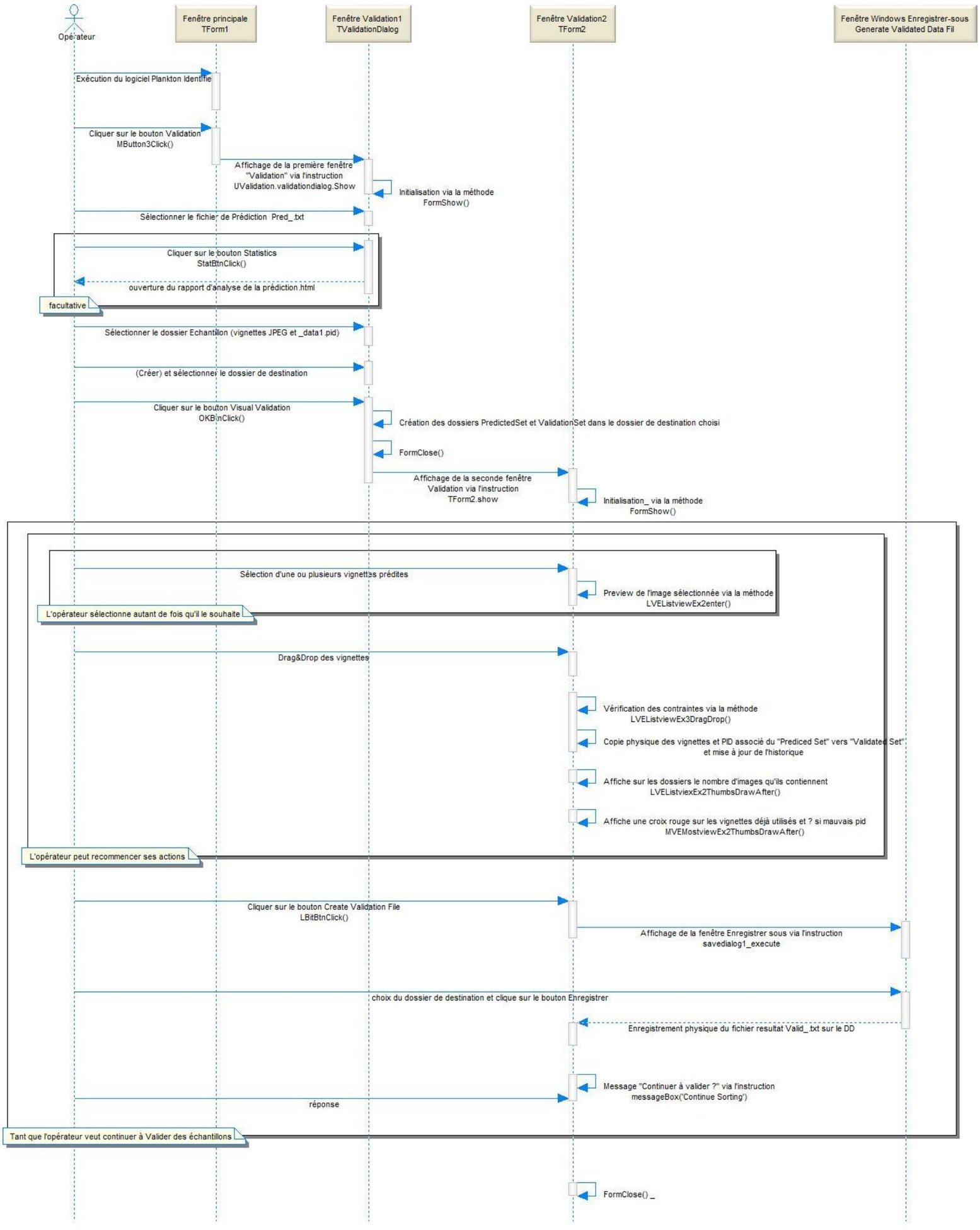

Opérateur
Fenêtre principale TForm1
Fenêtre Validation1 TValidationDialog
Fenêtre Validation2 TForm2
Fenêtre Windows Enregistrer-sous Generate Validated Data Fil
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Validation MButton3Click()
Affichage de la première fenêtre "Validation" via l'instruction UValidation.validationdialog.Show
Initialisation via la méthode FormShow()
Sélectionner le fichier de Prédiction Pred_.txt
Cliquer sur le bouton Statistics StatBtnClick()
ouverture du rapport d'analyse de la prédiction.html
facultative
Sélectionner le dossier Echantillon (vignettes JPEG et _data1.pid)
(Créer) et sélectionner le dossier de destination
Cliquer sur le bouton Visual Validation OKBtnClick()
Création des dossiers PredictedSet et ValidationSet dans le dossier de destination choisi
FormClose()
Affichage de la seconde fenêtre Validation via l'instruction TForm2.show
Initialisation_ via la méthode FormShow()
Sélection d'une ou plusieurs vignettes prédites
Preview de l'image sélectionnée via la méthode LVEListviewEx2enter()
L'opérateur sélectionne autant de fois qu'il le souhaite
Drag&Drop des vignettes
Vérification des contraintes via la méthode LVEListviewEx3DragDrop()
Copie physique des vignettes et PID associé du "Prediced Set" vers "Validated Set" et mise à jour de l'historique
Affiche sur les dossiers le nombre d'images qu'ils contiennent LVEListviexEx2ThumbsDrawAfter()
Affiche une croix rouge sur les vignettes déjà utilisés et ? si mauvais pid MVEMostviewEx2ThumbsDrawAfter()
L'opérateur peut recommencer ses actions
Cliquer sur le bouton Create Validation File LBitBtnClick()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog1_execute
choix du dossier de destination et clique sur le bouton Enregistrer
Enregistrement physique du fichier resultat Valid_.txt sur le DD
Message "Continuer à valider ?" via l'instruction messageBox('Continue Sorting')
réponse
Tant que l'opérateur veut continuer à Valider des échantillons
FormClose() _

Fig. 23 : Diagramme de séquence de la fonction Validation *(avant modifications)*

VI. Développement

1. Réalisations indispensables

1.1 Résolution du problème de gestion de mémoire

Les tests réalisés sur la version1.3.0, ont confirmé qu'après quelques manipulations des vignettes (sélections, glisser-déposer, …) le message d'erreur suivant apparaissait : « Ressource système insuffisant ». Ces tests ont également révélé que ce message d'erreur apparaissait aussi bien sous *WINDOWS XP* que sous *WINDOWS 7* (ce que le MOA n'avait pas constaté).

Les mêmes tests appliqués sur une version antérieure du logiciel (version 1.2.6) n'ont abouti à aucun message d'erreur. Cela semblait indiquer que la version 1.3.0 avait une gestion du cache de mémoire des vignettes différente de la version 1.2.6. Cette dernière utilise des versions différentes des composants additionnels que celles de la version 1.3.0. Après comparaison des codes, il est apparu que la source du problème venait d'un composant additionnel appelé *VIRTUALSHEELTOOLVELVEX*.

Afin de corriger ce problème, la dernière version du composant *VIRTUALSHEELTOOLVELVEX* a été installée avec le composant *VIRTUALSHEELTOOL* utilisé dans la version 1.2.6.

1.2 Modification du nom de boutons d'actions

La modification des intitulés des boutons a été très simple à mettre en œuvre. Il a suffi simplement d'accéder à la boite d' « Inspecteur d'objets » de l'IDE, et de modifier la valeur de la propriété « Caption » (Fig.24).

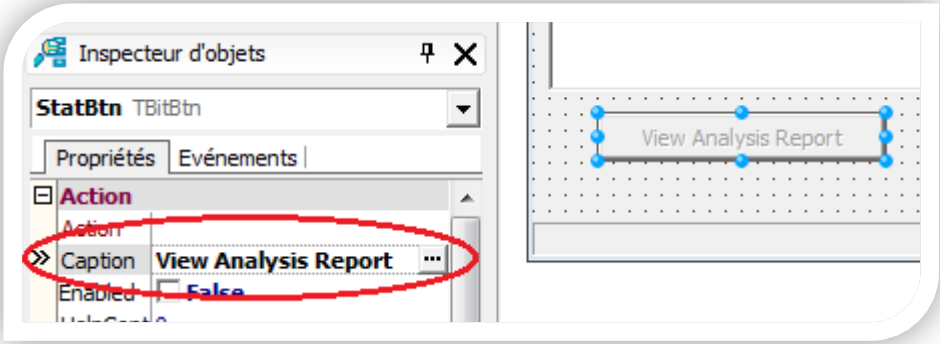


Fig. 24 : Exemple de modification du nom du bouton View Analysis Report

Les boutons suivants ont ainsi été modifiés dans les fonctions *Evaluation* et *Prediction* :

- « Statistics » en « View analysis report » ;
- « Start analysis » en « Run ».

1.3 Création des boutons "Dossier parent"

Revenir dans un dossier parent à partir d'un simple dérouleur est en effet assez fastidieux. Pour placer les boutons « Dossier parent » dans la fenêtre des fonctions *Learning* et *Validation*, il a d'abord fallu insérer le composant *TBitBtn* et lui attribuer un logo ( ), puis créer l'évènement « OnClick ». L'instruction pour modifier la valeur de la propriété *Path* du composant *LVEcombobox2* a été créée (Fig. 25) :

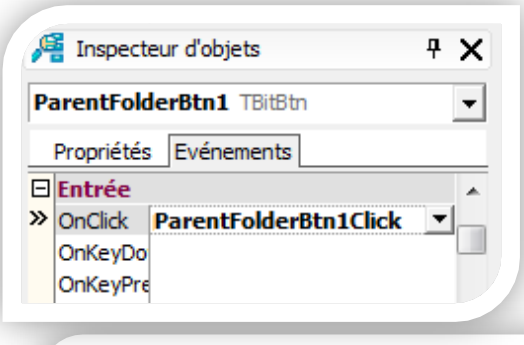


Fig. 25 : Code de l'évènement OnClick sur le bouton « Dossier parent »

```
procedure TFormLearning.ParentFolderBtn1Click(Sender: TObject);
begin
  LVEcombobox2.Path := ExpandFileName(LVEcombobox2.Path + '\..') ;
end;
```

Voici l'aperçu, pour la fenêtre *Learning,* avec les nouveaux boutons « Dossier parent » (Fig. 26).

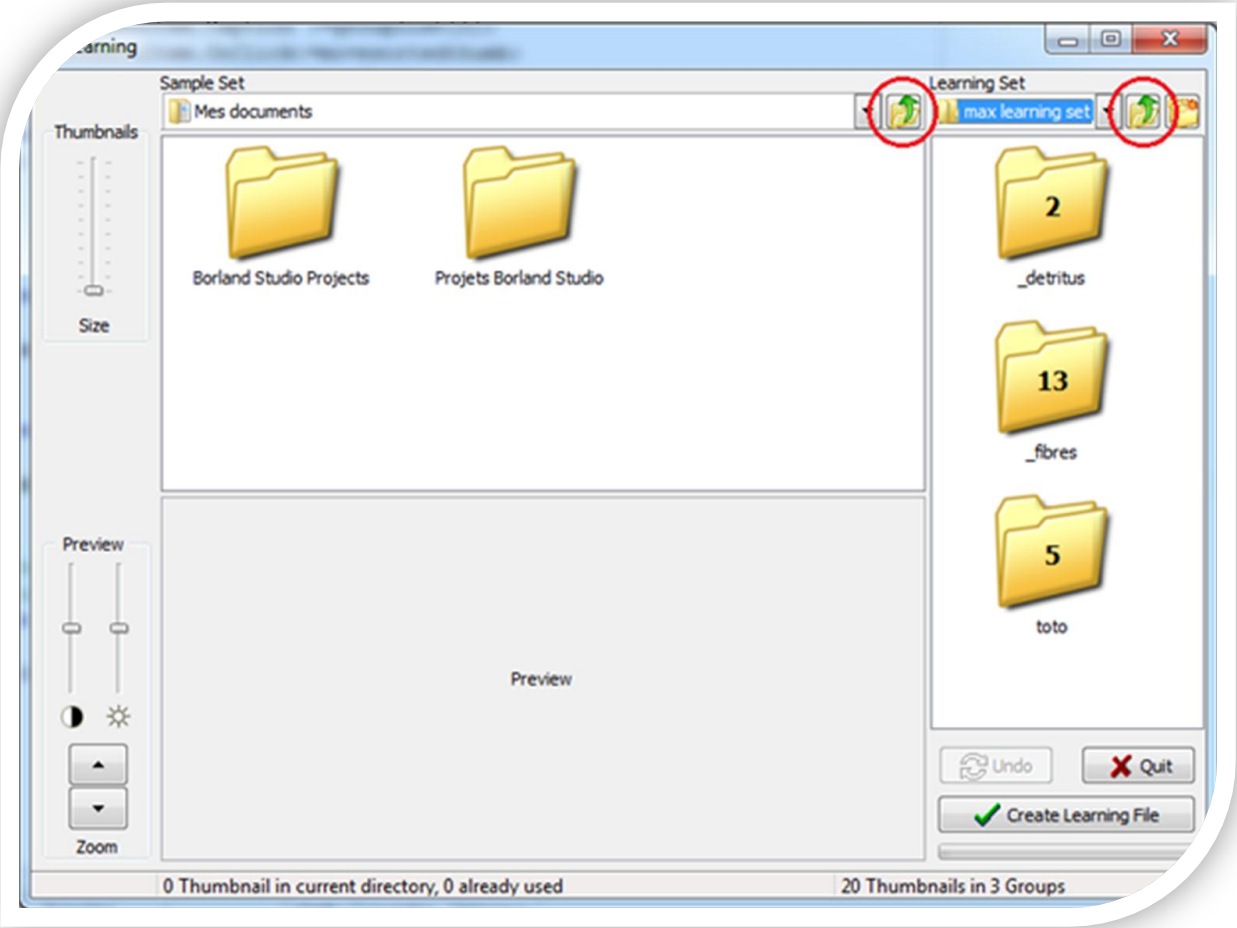


Fig. 26 : Aperçu de la fenêtre Learning avec les nouveaux boutons « Dossier Parent » encerclés en rouge

1.4 Modification de fonction du curseur Thumbnails

Afin de piloter à la fois la taille des vignettes et celles des dossiers dans les volets *Sample set* et *Learning set* via le curseur *Thumbnails* situé à gauche de la fenêtre *Learning*, les instructions pour modifier les valeurs des propriétés HEIGHT et WIDTH du composant *LVELISTVIEWEX3* ont été ajoutées dans le code. De plus, pour faire en sorte que la taille minimale soit plus petite, les coefficients multiplicateurs présents dans le calcul de la propriété WIDTH des composants *LVELISTVIEWEX2 et LVELISTVIEWEX3 ont été modifiés.*

avant

```
procedure TForm2.TrackBar1Change(Sender: TObject);
begin
  lvelistviewex2.ThumbsOptions.Height:=280-(20*trackbar1.Position);
  lvelistviewex2.ThumbsOptions.width:= 420-(30*trackbar1.Position);
end;
```

après

```
procedure TFormLearning.TrackBar1Change(Sender: TObject);
begin
  lvelistviewex2.ThumbsOptions.Height:=280-(20*trackbar1.Position);
  lvelistviewex2.ThumbsOptions.width:= 420-(33*trackbar1.Position);
  lvelistviewex3.ThumbsOptions.Height:=280-(20*trackbar1.Position);
  lvelistviewex3.ThumbsOptions.width:= 420-(33*trackbar1.Position);
end;
```

Fig. 27 : Code avant/après des instructions pour modifier les fonctions du curseur Thumbnails

Les modifications du composant *LVELISTVIEWEX3 ont* entrainé un problème de redimensionnement de l'icône des dossiers qui a été résolu en modifiant les valeurs des paramètres de l'appel de la méthode DRAW du composant ACANVAS dans l'évènement THUMBSDRAWBEFORE (Fig. 28 & 29) :

```
rocedure TFormLearning.LVEListviewEx3ThumbsDrawBefore(
  Sender: TCustomVirtualExplorerListviewEx; ACanvas: TCanvas;
  ListItem: TListItem; ThumbData: PThumbnailData; AImageRect,
  ADetailsRect: TRect; var DefaultDraw: Boolean);
var
 Path:string;
 folderbmp:tbitmap;
begin
  path:= LVEcombobox3.Path+'\'+listitem.Caption;
  if directoryexists(path) then begin
    DefaultDraw:=false;
    folderbmp:=tbitmap.Create;
    imagelist1.getbitmap(0,folderbmp);
    //Acanvas.Draw(20,0,folderbmp);
    Acanvas.Draw((AimageRect.Right-80)div 2,(AimageRect.Bottom-80)div 2,folderbmp);
    folderbmp.Free;
  end;
end;
```

Fig. 28 : Procédure de l'évènement ThumbsDrawBefore comprenant l'appel de méthode Draw avec les valeurs des paramètres à jour (en bleu)

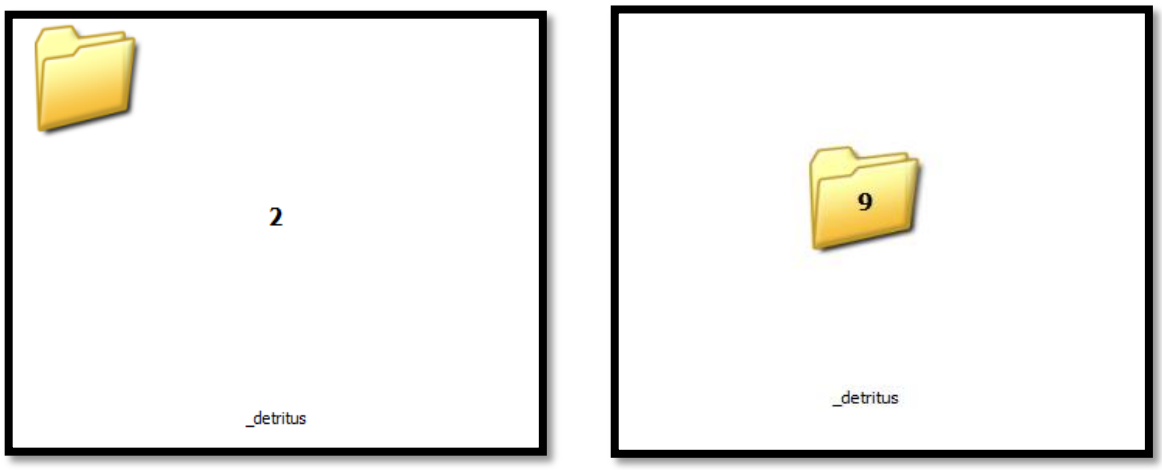


Fig. 29 : Redimensionnement des dossiers (avant/après)

1.5 Ouverture automatique du résultat html

Afin de faire apparaître directement le rapport d'analyse html produit par le logiciel TANAGRA en fin d'analyse dans les fonctions *Evaluation* et *Prediction*, le code suivant a été créé (Fig. 30) :

```
ReportPath:=OutputFolder+'\'+DestFile+'.html';
if fileexists(ReportPath) then begin
  sleep(700);
  ShellExecute(0,Nil,PChar(ReportPath),Nil,Nil,SW_SHOW);
end;
```

Fig. 30 : Code pour permettre l'ouverture automatique du fichier HTML

Ce code est présent dans la méthode de l'évènement *OnClick* du bouton « Run » et est situé juste après l'instruction permettant d'exécuter le logiciel *TANAGRA*.

Avant d'ouvrir le fichier HTML via la méthode *SHEELEXECUTE*, le programme test si les fichiers HTML existent.

1.6 Création du bouton « Edit tdm File »

Pour créer le bouton « Edit tdm File », le composant *TBITBTN*, a été inséré et un logo lui a été attribué. Un évènement « OnClick » a également été créé (Fig. 31) :

```
procedure TForm3.BtnEditTdmFileClick(Sender: TObject);
var
  methodfolder, name1, name2:string;
begin
  //preparer les fichers de configuration de Tanagra
  if evaluationmode then
    methodfolder:='\tdm_eval\'
  else
    methodfolder:='\tdm_class\';

  //chemin du fichier tdm à éditer
  name1:=InstallFolder+methodfolder+Dcombobox1.text;
  //ouverture via notepad.exe
  ShellExecute(HANDLE,PChar('open'),PChar('notepad.exe'), PAnsiChar(name1), nil,SW_SHOW);
end;
```

Fig. 31 : Procédure de l'évènement OnClick sur le bouton « Edit tdm File »

Un évènement *OnChange* (Fig. 32) du dérouleur situé à gauche du bouton « Edit tdm File », qui liste les différentes méthodes d'analyses, a été créé. Cet évènement permet de désactiver le bouton lorsque le dérouleur, nommé *DCOMBOBOX1*, est sur « Export to text file (no analysis) », car cette méthode n'est pas associé à un fichier tdm (Fig. 33).

```
procedure TForm3.DComboBox1Change(Sender: TObject);
begin
if Dcombobox1.text='Export to text file (no analysis)' then
  BtnEditTdmFile.Enabled:=False
else
  BtnEditTdmFile.Enabled:=True;
end;
```

Fig. 32 : Procédure de l'évènement OnChange du dérouleur DCOMBOBOX1

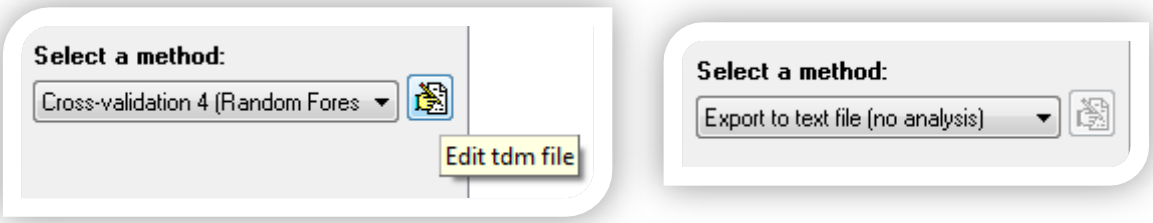


Fig. 33 : Bouton « Edit tdm File » activé/désactivé

1.7 Modification du nom des fichiers pré-sauvegardés

Le code pour modifier les noms des fichiers pré-sauvegardés a été modifié comme indiquer dans la figure 34. En fonction de l'option choisie (*Evaluation* ou *Prediction*), la fenêtre *Enregistrer-sous* propose de nommer le fichier de résultat Pred_xxxx ou Eval_xxxx :

```
  savedialog2.InitialDir:=OutputFolder+'\';
// En fonction du mode
if evaluationmode then
savedialog2.FileName:= OutputFolder+'\Eval_'+RunIndex
else
savedialog2.FileName:= OutputFolder+'\Pred_'+RunIndex;
```

Fig. 34 : Initialisation du champ de texte de la fenêtre « Enregistrer-sous »

1.8 Suppression de la création de dossiers

Les instructions permettant de créer les dossiers « PredictedSet » et « ValidatedSet » dans la fonction *Validation* ont été supprimées dans le code source.

1.9 Création d'une checkbox « Keep unsorted thumbnails »

La création d'une checkbox « Keep unsorted thumbnails » dans la fenêtre de sélection de la fonction *Validation* (cf. Fig. 8) va permettre de supprimer les vignettes non triées qui ont été copiées dans le dossier de destination ou au contraire de les garder. La case est activée par défaut. Le code de l'instruction qui teste si la case est coché ou non et qui appelle la méthode « DeleteFile » lorsque cette case est décochée est présenté figure 35.

```
if not CheckBoxKeepUnsortedThumbnails.checked then
  DeleteFile(DirList.Strings[I]+'\'+lignedata.Strings[LabelIndex]+'_'
     +lignedata.Strings[ItemIndex]+'.jpg');
end;
```

Fig. 35 : Appel de la méthode « DeleteFile »

Le nouvel aperçu de la fenêtre *Validation* est présenté figure 36 :

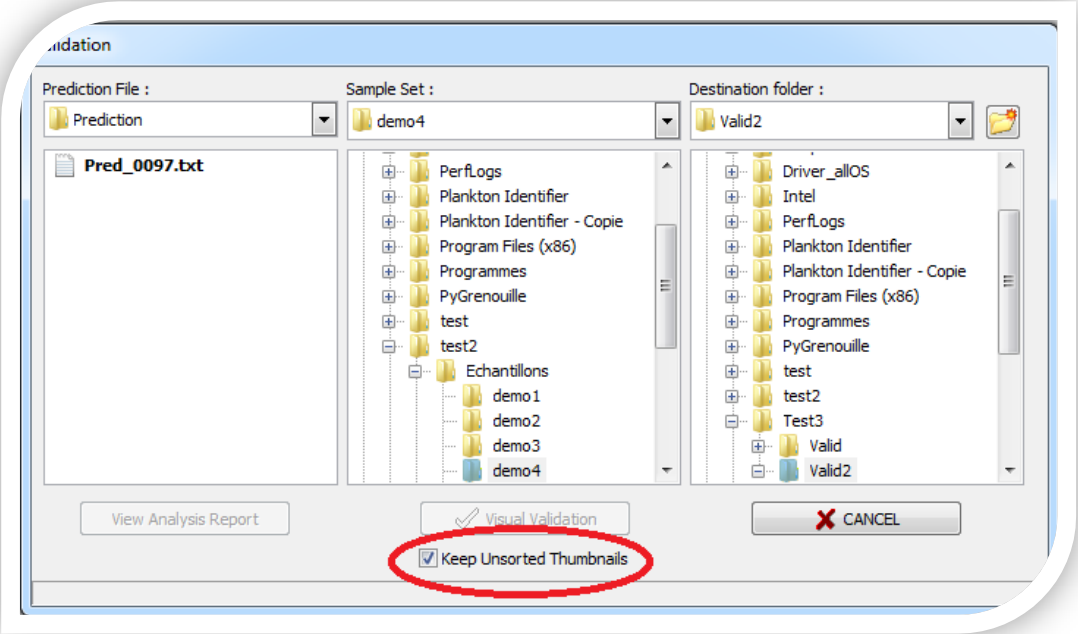


Fig. 36 : Aperçu de la fenêtre de sélection de la fonction Validation

1.10 Améliorations de la fonction Validation

La fonction *Validation* a besoin d'être améliorée en distinguant une partie « Création d'une validation à partir d'un fichier de prédiction » et une partie « Re-vérification d'une validation », et a besoin d'être améliorée en intégrant l'option « Move to » à la place de l'option « Glisser-déposer » dans la dernière fenêtre.

Une nouvelle fenêtre a donc été créée, et propose désormais, après avoir cliqué sur le bouton « Validation » depuis le menu principal, deux choix : la création d'une validation à partir d'un fichier de prédiction ou la re-vérification des vignettes validées avec possibilité de modification (Fig. 37).

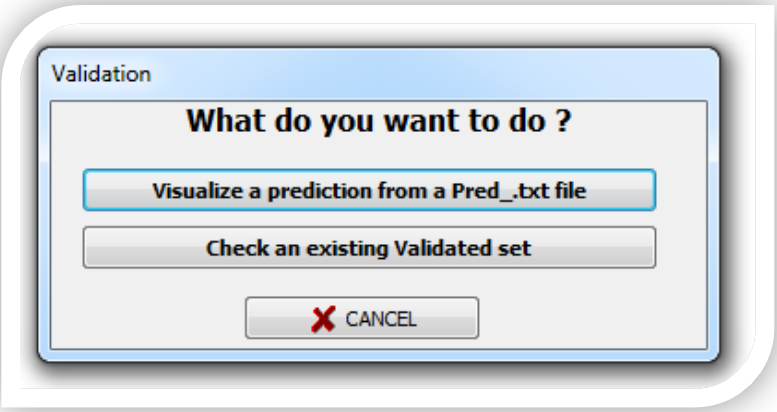


Fig. 37 : Nouvelle fenêtre de l'option Validation

1.10.1 Validation à partir d'une prédiction

Le clic sur la première possibilité ouvre la fenêtre de sélection (cf. Fig. 8) qui reste inchangée. La fenêtre qui suit après avoir cliqué sur le bouton « Visual validation », a, quant à elle, été revue (Fig. 38).

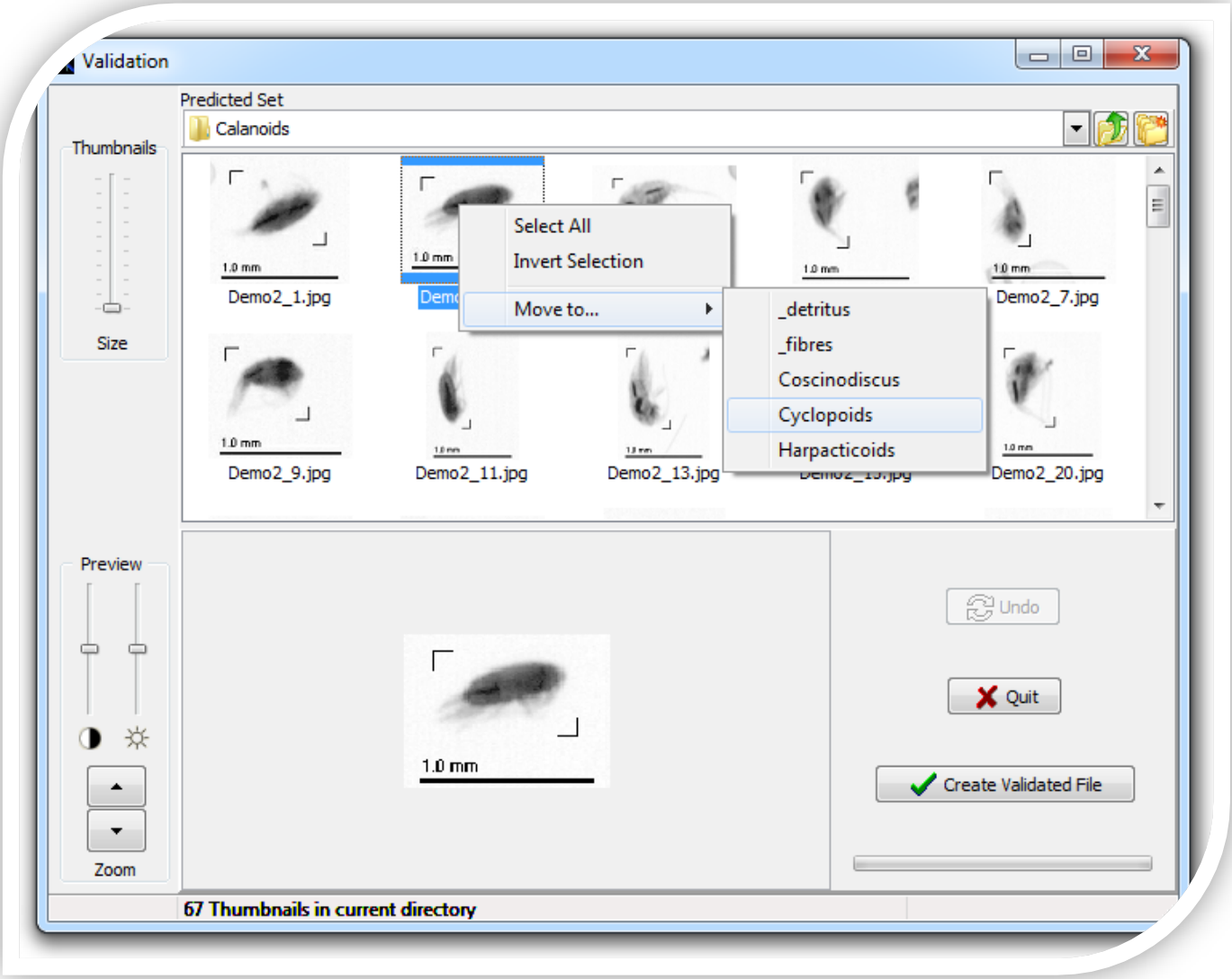


Fig. 38 : Mise à jour de la fenêtre Validation

La possibilité de Glisser-déposer les vignettes a été remplacée par l'option « Move to » selon les recommandations des maîtres d'ouvrage.

L'option « Delete » a été optimisée et est disponible seulement lorsqu'un ou plusieurs dossiers sélectionnés ne contiennent pas de vignettes JPEG (et cela même s'ils contiennent un fichier « nom_echantillon_dat1.pid »). Un message « Delete is not allowed » apparaît quand on essaye de supprimer une vignette ou un dossier

contenant au moins une vignette depuis le bouton de suppression du clavier ou depuis les touches de raccourcies du « couper » (CTRL+X).

1.10.2 Re-vérification d'un échantillon validé

Le clic sur la deuxième possibilité (« Check an existing Validated set ») ouvre la fenêtre de sélection présentée en figure 39.

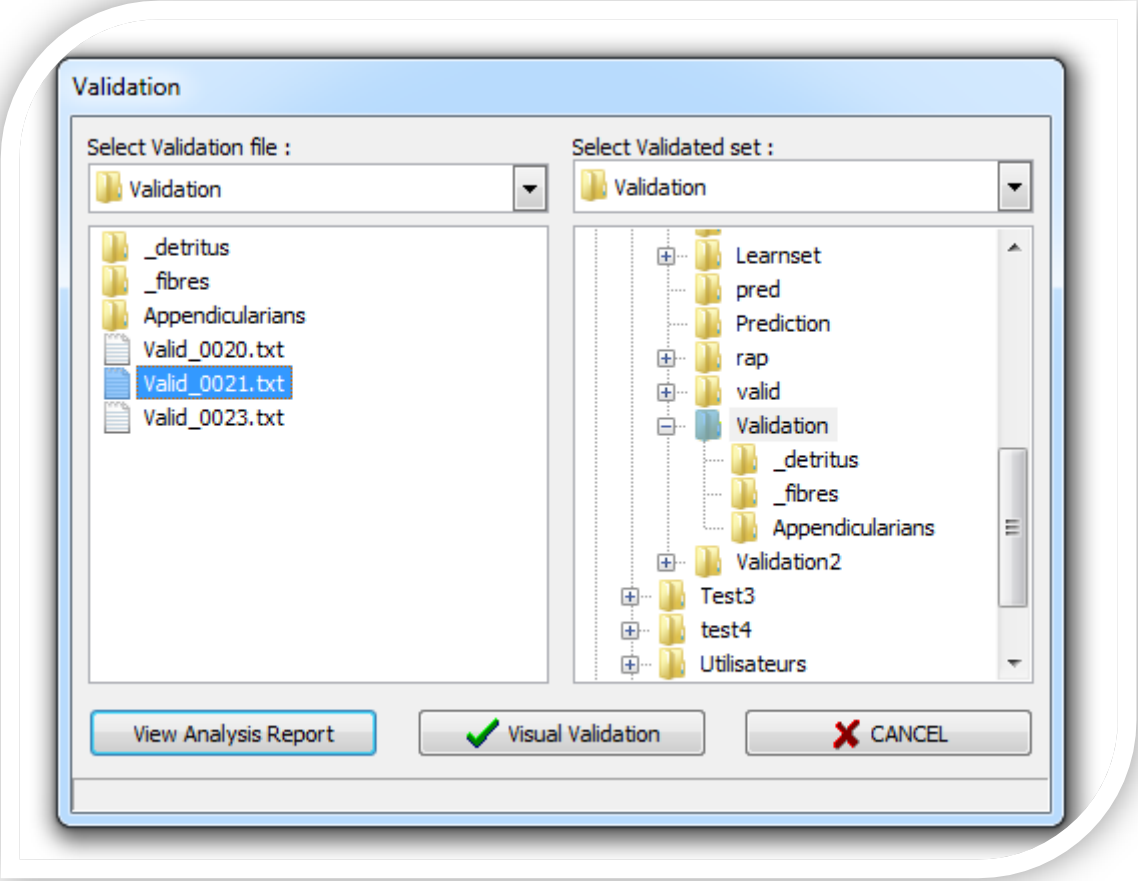


Fig. 39 : Nouvelle fenêtre de sélection

Dans cette fenêtre, l'opérateur doit sélectionner non pas un fichier « Pred_xxxx.txt » mais un fichier « Valid_xxxx.txt » avec le dossier « Validated set » correspondant ». Cliquer sur le bouton « Visual validation » redirige l'opérateur vers la deuxième fenêtre de validation (cf. Fig. 38). L'option « Move to » liste les groupes d'identification grâce au nom des dossiers du « Validated set ». La création du nouveau fichier de validation, suite à la re-vérification, mettra à jour la dernière colonne « Valid » du fichier de validation.

2. Diagrammes de séquences après les modifications

Après avoir réalisé les modifications du logiciel *PLANKTON IDENTIFIER*, les diagrammes de séquences et le modèle opérationnel de traitements présentés dans le chapitre « Analyse de l'existant », ont été mises à jour.

Le diagramme de séquences de la fonction *Validation* pour créer une validation à partir d'une prédiction et le diagramme de séquences de la fonction Validation pour re-vérifier une validation sont respectivement présentés ci-dessous (Fig. 40 & 41). Les autres diagrammes et le modèle opérationnel de traitements, qui ont subis légèrement des modifications, sont présentés en **annexe VI**, **VII** et **VIII**.

3. Réalisations optionnelles

3.1 Amélioration du logiciel PID Viewer

Afin de rendre éditable l'onglet *Header,* la valeur de la propriété *READONLY* du composant *RICHEDIT1* (celui qui affiche la partie *Header* du fichier PID) a été modifiée en *False*.

Ensuite, il a fallu insérer un bouton « Save » et lui attribuer un logo, puis créer l'évènement « OnClick ». La procédure d'exécution de cet évènement est présentée ci-dessous (Fig. 42) :

Diagramme de séquence - après MAJ		
La fonction Validation : Création du Valid_ .txt à partir du Pred_ .txt		
Auteur : LEDEE		
Version : 1.4	Créé le : 18/4/2013	Modifié le : 30/5/2013

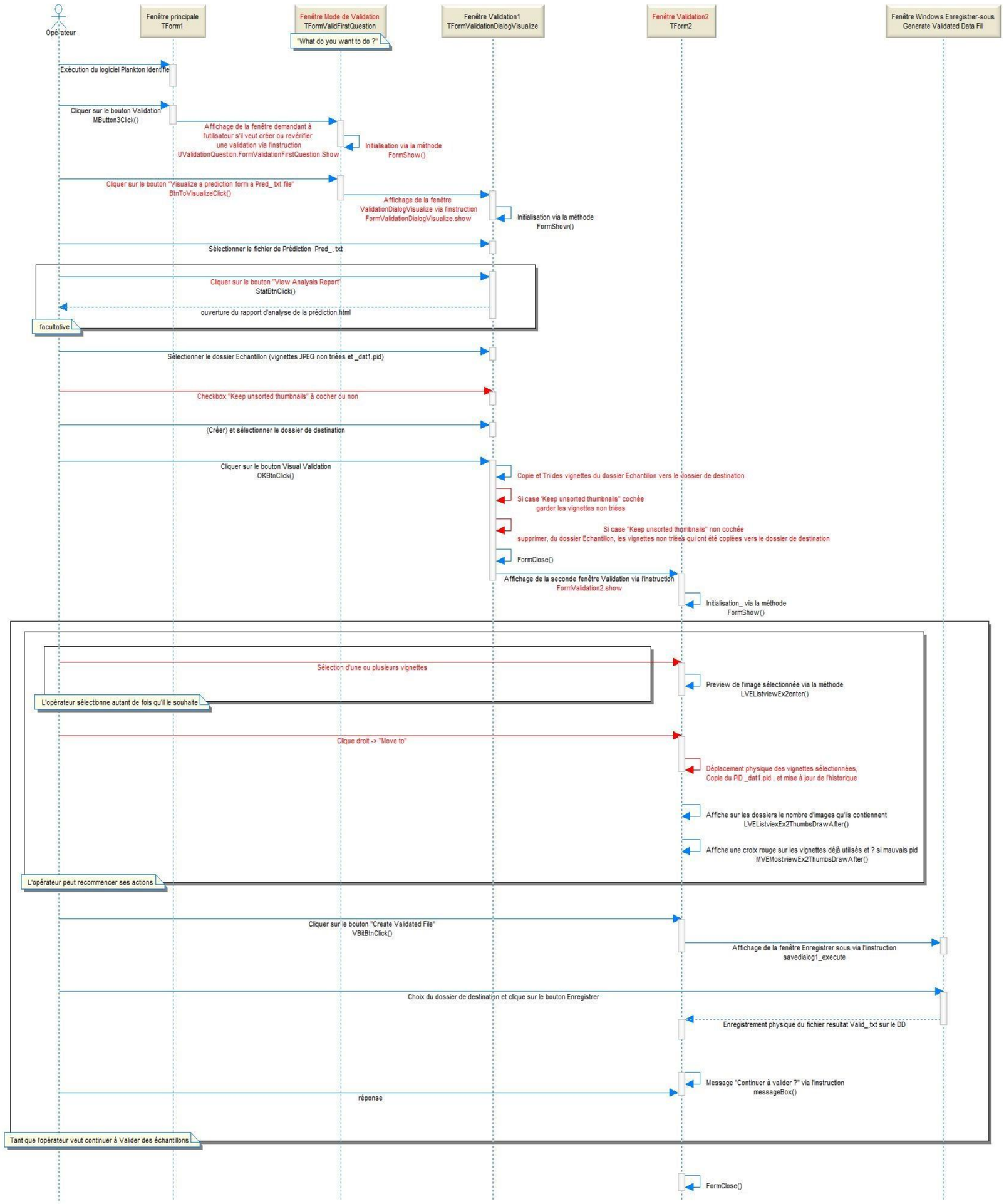

Opérateur
Fenêtre principale TForm1
Fenêtre Mode de Validation TFormValidFirstQuestion
"What do you want to do ?"
Fenêtre Validation1 TFormValidationDialogVisualize
Fenêtre Validation2 TForm2
Fenêtre Windows Enregistrer-sous Generate Validated Data Fil
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Validation MButton3Click()
Affichage de la fenêtre demandant à l'utilisateur s'il veut créer ou revérifier une validation via l'instruction UValidationQuestion.FormValidationFirstQuestion.Show
Initialisation via la méthode FormShow()
Cliquer sur le bouton "Visualize a prediction form a Pred_.txt file" BtnToVisualizeClick()
Affichage de la fenêtre ValidationDialogVisualize via l'instruction FormValidationDialogVisualize.show
Initialisation via la méthode FormShow()
Sélectionner le fichier de Prédiction Pred_.txt
Cliquer sur le bouton "View Analysis Report" StatBtnClick()
ouverture du rapport d'analyse de la prédiction.html
facultative
Sélectionner le dossier Echantillon (vignettes JPEG non triées et _dat1.pid)
Checkbox "Keep unsorted thumbnails" à cocher ou non
(Créer) et sélectionner le dossier de destination
Cliquer sur le bouton Visual Validation OKBtnClick()
Copie et Tri des vignettes du dossier Echantillon vers le dossier de destination
Si case 'Keep unsorted thumbnails" cochée garder les vignettes non triées
Si case "Keep unsorted thumbnails" non cochée supprimer, du dossier Echantillon, les vignettes non triées qui ont été copiées vers le dossier de destination
FormClose()
Affichage de la seconde fenêtre Validation via l'instruction FormValidation2.show
Initialisation_ via la méthode FormShow()
Sélection d'une ou plusieurs vignettes
Preview de l'image sélectionnée via la méthode LVEListviewEx2enter()
L'opérateur sélectionne autant de fois qu'il le souhaite
Clique droit -> "Move to"
Déplacement physique des vignettes sélectionnées, Copie du PID _dat1.pid , et mise à jour de l'historique
Affiche sur les dossiers le nombre d'images qu'ils contiennent LVEListviexEx2ThumbsDrawAfter()
Affiche une croix rouge sur les vignettes déjà utilisés et ? si mauvais pid MVEMostviewEx2ThumbsDrawAfter()
L'opérateur peut recommencer ses actions
Cliquer sur le bouton "Create Validated File" VBitBtnClick()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog1_execute
Choix du dossier de destination et clique sur le bouton Enregistrer
Enregistrement physique du fichier resultat Valid_.txt sur le DD
Message "Continuer à valider ?" via l'instruction messageBox()
réponse
Tant que l'opérateur veut continuer à Valider des échantillons
FormClose()

Fig. 40 : Diagramme de séquence de la fonction Validation *pour créer une validation à partie d'une prédiction*

Diagramme de séquence - après MAJ

La fonction Validation : Vérification d'une Validation

Auteur : LEDEE

Version : 1.4 Créé le : 18/4/2013 Modifié le : 29/5/2013

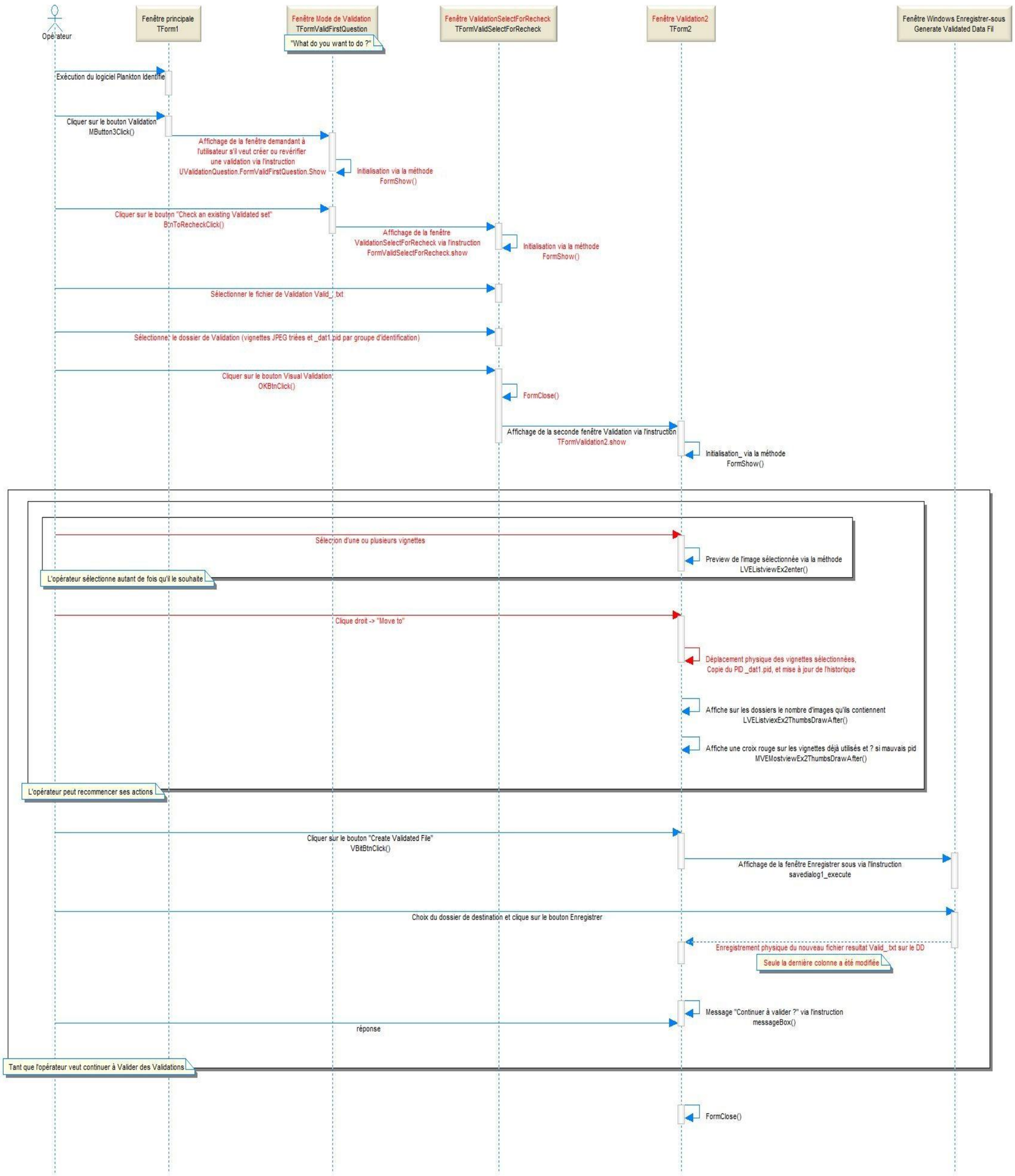

Opérateur
Fenêtre principale TForm1
Fenêtre Mode de Validation TFormValidFirstQuestion
"What do you want to do ?"
Fenêtre ValidationSelectForRecheck TFormValidSelectForRecheck
Fenêtre Validation2 TForm2
Fenêtre Windows Enregistrer-sous Generate Validated Data Fil
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Validation MButton3Click()
Affichage de la fenêtre demandant à l'utilisateur s'il veut créer ou revérifier une validation via l'instruction UValidationQuestion.FormValidFirstQuestion.Show
Initialisation via la méthode FormShow()
Cliquer sur le bouton "Check an existing Validated set" BtnToRecheckClick()
Affichage de la fenêtre ValidationSelectForRecheck via l'instruction FormValidSelectForRecheck.show
Initialisation via la méthode FormShow()
Sélectionner le fichier de Validation Valid_.txt
Sélectionner le dossier de Validation (vignettes JPEG triées et _dat1.pid par groupe d'identification)
Cliquer sur le bouton Visual Validation OKBtnClick()
FormClose()
Affichage de la seconde fenêtre Validation via l'instruction TFormValidation2.show
Initialisation_ via la méthode FormShow()
Sélection d'une ou plusieurs vignettes
Preview de l'image sélectionnée via la méthode LVEListviewEx2enter()
L'opérateur sélectionne autant de fois qu'il le souhaite
Clique droit -> "Move to"
Déplacement physique des vignettes sélectionnées, Copie du PID _dat1.pid, et mise à jour de l'historique
Affiche sur les dossiers le nombre d'images qu'ils contiennent LVEListviexEx2ThumbsDrawAfter()
Affiche une croix rouge sur les vignettes déjà utilisés et ? si mauvais pid MVEMostviewEx2ThumbsDrawAfter()
L'opérateur peut recommencer ses actions
Cliquer sur le bouton "Create Validated File" VBitBtnClick()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog1_execute
Choix du dossier de destination et clique sur le bouton Enregistrer
Enregistrement physique du nouveau fichier resultat Valid_.txt sur le DD
Seule la dernière colonne a été modifiée
Message "Continuer à valider ?" via l'instruction messageBox()
réponse
Tant que l'opérateur veut continuer à Valider des Validations
FormClose()

Fig. 41 : Diagramme de séquence de la fonction Validation *pour re-valider une validation*

```
procedure TMainForm.ToolButton2Click(Sender: TObject);
var
  index,i,nbBackup:Integer;
  Child: TMDIChild;
  Txt, ligne, NameNewBackupFile: String;
  OldPID, NewPID:Textfile;
  FF: TSearchRec;

begin
  Txt:='';
  NameNewBackupFile:='';
  index:=0;
  // Je cherche le child sélectionné pour enregistrer le nouveau .pid
  while (not Mainform.MDIChildren[index].active) and (index<(Mainform.MDIChildCount-1)) do
    index:=index+1;
  Child:=TMDIChild(Mainform.MDIChildren[index]);

  if FileExists(Child.GetSrcPath) then begin
     // recherche pour récupérer le dernier numero
     nbBackup:=0;
     // S'il trouve des sauvegardes .pid.~
     if FindFirst(ExtractFilePath(Child.GetSrcPath)+'\'+ExtractFileName(Child.GetSrcPath)+'.~*',faArchive,FF)=0 then begin
         repeat
            nbBackup:=nbBackup+1;
         until findnext(FF)<>0;
     end;
     FindClose(FF);

     // S'il n'existe pas déjà de fichier Backup portant le chiffre  nbBackup+1 alors on copie l'original et on renomme le new fichier
     if not FileExists(Child.GetSrcPath+ '.~' + IntToStr(nbBackup+1)) then begin
          NameNewBackupFile:=Child.GetSrcPath+ '.~' + IntToStr(nbBackup+1);       // = .pid~1
          RenameFile(Child.GetSrcPath, NameNewBackupFile);
     end
     // Sinon, on rajoutte +n jusqu'à ce que le fichier Backup portant le chiffre  nbBackup+n   n'existe pas , puis on copie l'original et on renomme le new fichier
     else begin
        nbBackup:=nbBackup+1;
        while FileExists(Child.GetSrcPath+ '.~' + IntToStr(nbBackup)) do begin
            nbBackup:=nbBackup+1;
        end;
               NameNewBackupFile:= Child.GetSrcPath+ '.~' + IntToStr(nbBackup);
               RenameFile(Child.GetSrcPath, NameNewBackupFile);
     end;

   // Création du NOUVEAU .PID modifié
    AssignFile(NewPID, Child.GetSrcPath);
    ReWrite(NewPID); // Créer le new .PID modifié

   // COPIE de la nouvelle partie HEADER vers NEW PID
    Child.RichEdit1.PlainText:=true;
    for i := 0 to Child.RichEdit1.Lines.Count -1 do begin
        //Delete(NewPID, i) ;
        WriteLn(NewPID, Child.RichEdit1.Lines.Strings[i]);
    end;
    Child.RichEdit1.PlainText:=false;

   // Ouverture en lecture seule de l'ancien PID .pid~1 pour copier la partie DATA
    AssignFile(OldPID, NameNewBackupFile);
    Reset(OldPID);
    ligne:='';
    while ligne <>'[Data]' do
      Readln(OldPID, ligne);
    WriteLN(NewPID, ligne) ; // Ecriture de la ligne [Data]

    repeat                 // Je li et je copie les lignes entre [Data] et jusque la fin du fichier du OLD PID vers NEW PID
        Readln(OldPID, ligne);
        WriteLN(NewPID, ligne) ;
    until eof(OldPID);

    CloseFile(OldPID);
    CloseFile(NewPID);

  end;
  MainForm.ToolButton2.enabled:=false;
  ShowMessage('New backup file created : ' + NameNewBackupFile);
end;
```

Fig. 42 : Code d'exécution de l'évènement OnClick du bouton Save

Le bouton « Save » étant désactivé par défaut, l'évènement « *OnSelectionChange* » du composant *RICHEDIT1* a été créé pour activer ce bouton une fois l'édition commencée. L'aperçu du logiciel après édition de la partie *Header* est présenté en figure 43.

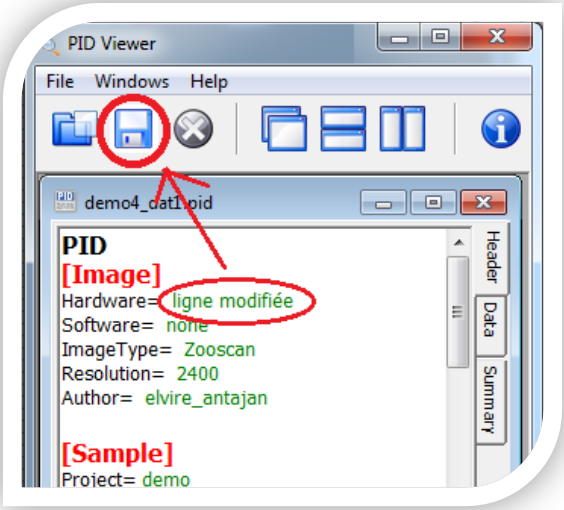


Fig. 43 : Bouton « Save » activé après édition de la partie Header

L'aperçu du dossier contenant le fichier PID modifié et l'ancien fichier PID après avoir cliqué sur le bouton « Save » est présenté ci-dessous (Fig. 44) :

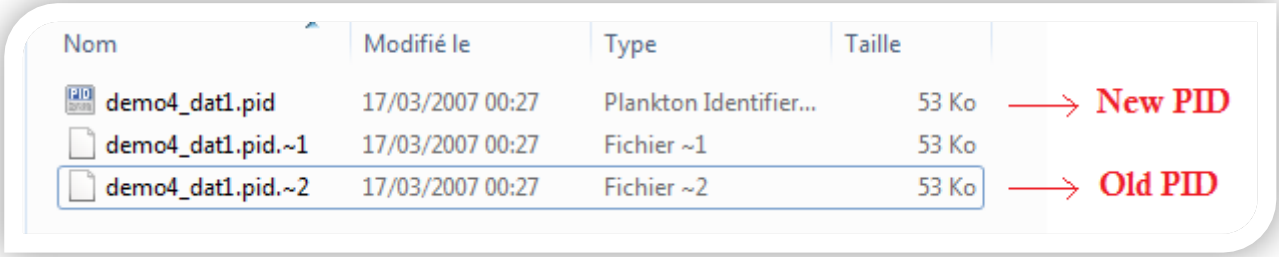


Fig. 44 : Aperçu du dossier contenant le nouveau et l'ancien fichier PID

3.2 Création de maquettes de fenêtres pour la future fonction Compilation

Faute de temps, la nouvelle fonction *Compilation* n'a pu être réalisée, néanmoins des maquettes des fenêtres de cette nouvelle fonction ont été proposées au MOA. La maquette qui modélise la fenêtre *Compilation* après avoir cliqué sur le bouton « *Compilation* » à partir du menu principal est la suivante (Fig. 45) :

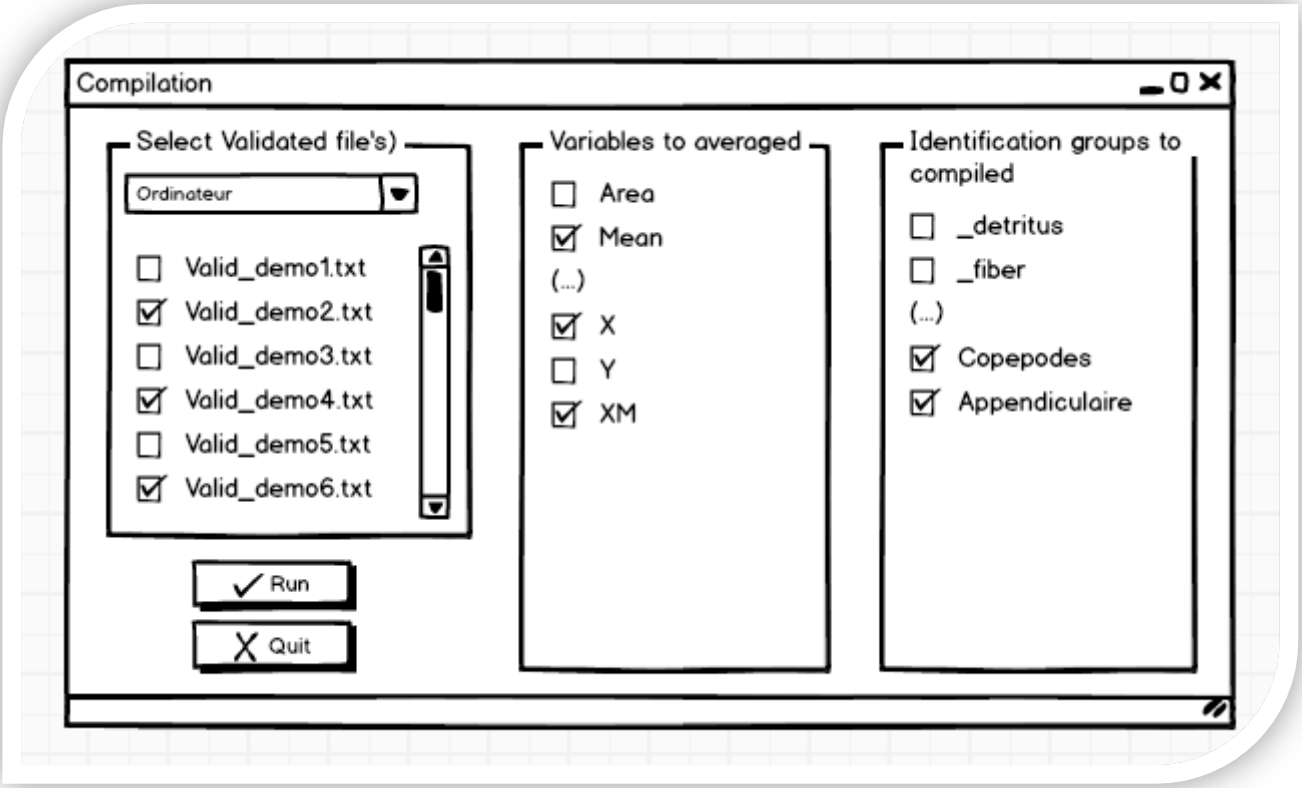


Fig. 45 : Maquette de la fenêtre Compilation

Après avoir sélectionné les fichiers de validation, les variables, et les groupes d'identification et après avoir cliqué sur le bouton « Run », un nouveau fichier nommé « Compil_xxxx.txt » est créé. Il contiendra les résultats présentés ci-dessous (Fig. 46).

Compil_SampleName.txt

SampleName	GroupName	Counts	SubPart	FracId	Area	Mean	...	ESD
Demo1	Copepodes	158	4	...	...	...	...	...
Demo1	Appendiculaires	13	4	..	...	...	...	...
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
Demo2	Copepodes	97	16	...	...	...	...	...
Demo2	Appendiculaires	3	16	...	...	...	...	...
...	...	...	...	...	...	...	...	...

Fig. 46 : Maquette du contenu du fichier de Compilation

VII. Perspectives d'évolutions du logiciel

1. Tests unitaires

Le planning prévisionnel prévoyait de réaliser des tests unitaires documentés dans le but de vérifier le bon fonctionnement des parties précises du logiciel *PLANKTON IDENTIFIER*. Même si les tests ont été effectués rigoureusement, les traces écrites n'ont pas été rédigées. Après en avoir discuté avec les maîtres d'ouvrage, il était plus important d'améliorer le logiciel *PID VIEWER* que de réaliser des tests unitaires documentés. Ces tests pourront être plus approfondis par les futurs développeurs pour la prochaine mise à jour du logiciel afin de déceler certaines anomalies et les corrigées.

2. Bug focus

N'ayant pas trouvé de solutions pour corriger le bug du focus lors du survol de la souris sur un dossier, nous soupçonnons que la source du problème soit liée à l'un des composants utilisé. Il faudrait procéder à des tests en ce sens, ce qui demanderait beaucoup de temps. Ce bug ne remettant pas en cause le bon fonctionnement du logiciel *PLANKTON IDENTIFIER,* les maîtres d'ouvrage n'ont pas souhaité que j'aille plus loin dans ma démarche.

3. Visualisation des vignettes lors d'une multi-sélection

Après avoir testé les démonstrations des composants additionnels fournies par les concepteurs, ce problème était également présent. Pour résoudre ce problème, il est conseillé de remplacer le composant affecté par un autre.

4. Fonction Compilation

Pour faute de temps, la fonction *Compilation* n'a pas pu être développée mais des maquettes, présentées dans le chapitre précédent, ont été créées pour les autres futurs développeurs.

5. Changement d'IDE

L'IDE *Borland Studio Developer 2006* n'étant plus en vente ni même maintenu à jour, le changement d'un IDE est fortement recommandé. Il est conseillé d'utiliser soit une version d'IDE *Delphi* conçue et maintenue à jour par la société *Embarcadero* ou soit *WinDev* qui est très complète et très rapide à prendre en main mais qui nécessite de redévelopper le logiciel avec le langage *WLangage*.

VIII. Compétences acquises

Durant ce projet, j'ai dû m'impliquer au maximum dans l'organisation de mon travail, comprendre exactement ce que je faisais et savoir pourquoi et comment je le faisais et ce au sein d'une équipe.

De cette expérience, j'ai retenu qu'être autonome implique de devoir prendre des initiatives pour planifier et exécuter des tâches, de faire des bilans, d'évaluer son efficacité, de réfléchir aux méthodes utilisées et de faire des propositions d'optimisations du cahier des charges aux maîtres d'ouvrage. La réflexion est essentielle dans l'acquisition de l'autonomie.

J'ai ainsi su raisonner avec logique et travailler avec analyse ; la lecture de documentations techniques et scientifiques m'a permis de comprendre le système. J'ai également su communiquer avec des utilisateurs non informaticiens pour bien comprendre leurs attentes. Dans ce sens, la mise en place du blog a été un outil d'échange intéressant entre moi et les maîtres d'ouvrage. Toutefois, l'expérience montre qu'il est nécessaire de bien expliquer en amont l'utilité de ce genre d'outil qui n'a pas forcément habituel pour les maîtres d'ouvrage par rapport aux échanges de mails.

Conclusions

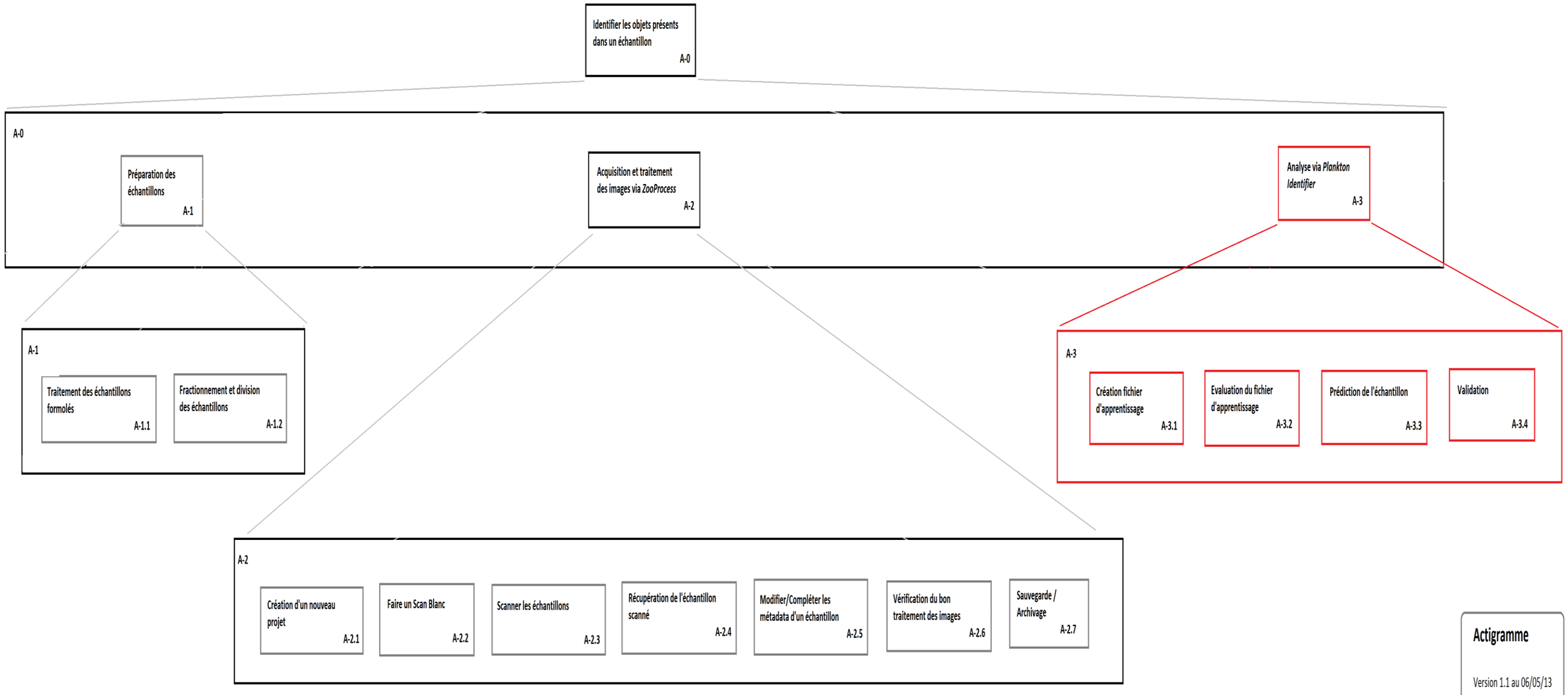
J'ai pu, durant ce stage au centre Ifremer de Boulogne-sur-Mer, mettre à profit mes acquis informatiques pour répondre aux besoins qui m'ont été définis à savoir modifier et améliorer le logiciel *PLANKTON IDENTIFIER* dans le but d'optimiser son utilisation. Mon diplôme DEUST Technicien de la mer et du littoral obtenu en 2010 a facilité la compréhension du contexte.

Ce projet de fin d'étude m'a permis d'améliorer ma méthode de travail, d'accroitre mes connaissances dans le développement informatique et aussi dans la gestion de projet. Pour répondre pleinement à la demande des maîtres d'ouvrage, il a fallu accepter de faire évoluer le cahier des charges ce qui a constitué une difficulté supplémentaire et demander un esprit d'adaptation. Lorsqu'un problème se présentait, je n'ai pas hésité à proposer mes solutions. L'autonomie et l'écoute ont été des éléments déterminant pour mettre à bien ce projet.

Je tire donc un bilan très positif de cette expérience et cela confirme mon désir de poursuivre mes études dans le domaine de la bio-informatique notamment grâce à ma double compétence en biologie et en informatique.

ANNEXES

<u>Annexe I</u> : Actigramme du processus d'identification d'un échantillon



Annexe II : Contenu d'un fichier PID d'un échantillon

PID
[Image]
Hardware= none
Software= none
ImageType= Zooscan
Resolution= 2400
Author= elvire_antajan

[Sample]
Project= demo
SampleId= demo2
Scanop= elvire_antajan
Ship= none
Scientificprog= zoopnec
StationId= none
Date= 20010820-9999
Latitude= 99.99
Longitude= 88.88
Depth= 9999
CTDref= none
Otherref= none
Townb= 1
Towtype= 2
Nettype= WP2
Netmesh= 200
Netsurf= 0.26
Zmax= 9999
Zmin= 0
Vol= 9999

[Fraction]
Code= A
FracId= A
Min= 200
Max= 999

[Subsample]
SubPart= 9999
SubMethod= motoda
CellPart= 1
Replicates= 1
VolIni= 1
VolPrec= 1
Observation= none

[Process]
ConfigFile= CNRS_bcz_2003_config.txt
Version= 02/06/2006_Picheral_CNRS
Multiple_image_process= YES
Date= 20060616_1559

[Image_Process]
vis1_file= PROCESS
Date= 20060616_1559
01dec03demo2
Image_process= Zooscan_1asep
Version= 09/06/2006_Picheral_CNRS
Image_path= E:\\Zooscan_bcz
Image= 01dec03demo2.tif
Frame= _1
Raw_image_depth= 16 bits
Raw_image_size= 257536338
Raw_Image_BackUp_as_ZIP= NO

Final_Image= 8bits
Corrected_width= 7130
Corrected_heigth= 17620
Grey_Level_Adjustement= YES
ODmethod= CNRS2003
Doxabspos= 3700
Doyabspos= 17000
ODxminref= 3400
ODyminref= 17410
ODxdimref= 600
ODydimref= 160
OD_measured= 144.5827
Max-min_grey_difference_in_OD= 37
STD= 3.3179
OD_Settings= 150
OD_corrected= 149.5827
Rolling_ball_background_subtrack_value= 300
Vis_image_save= YES

[Particules_Process]
Version= 23/05/2006_Picheral_CNRS
Resolution= 2400
Calibration= Calibrage_CNRS
Threshold= 248.0
PixSize= 0.0106
MinSize= 0.3173
MaxSize= 3.1596
Method= default [0.3000 - 3]
Measures_unit= pixel
Min_Area= 706
Max_Area= 70002
Separation_Mask= UNUSED
Particle_Analysis= YES
Analysis_Method= from_thumbnail_images_fractalA_skeleton
Analysed_Objects= 251
TAG= NO
Results= SAVED
Date_end= 20060616_1626

[Data]
!Item;Label;Area;Mean;StdDev;Mode;Min;Max;X;Y;XM;YM;Perim.;BX;BY;Width;Height;Major;Minor;Angle;Circ.;Feret;IntDen;Median;Skew;Kurt;%Area;XStart;YStart;Area_exc;Fractal;Skelarea
1;demo2;2407;175.97;71.03;248;20;252;43.34;31.44;42.89;30.37;423.71;1412;1533;78;60;73.25;41.84;32.96;0.17;85;423564;204;-0.58;-1.12;1.25;1461;1533;2377;1.22;454
2;demo2;2184;173.87;66.84;248;25;252;40.01;22.77;38.6;24.21;297.83;1017;1776;79;47;77.94;35.68;18.29;0.31;83.17;379731;199;-0.47;-1.3;0.37;1066;1776;2176;1.15;346
3;demo2;3236;189.27;48.91;248;43;250;36.06;28.91;37.75;28.78;369.1;5835;2338;112;70;83.64;49.26;30.96;0.3;114.98;612488;198;-0.76;-0.2;0.06;5873;2338;3234;1.14;254
4;demo2;1877;224.96;16.06;230;135;248;24.61;25.4;24.76;25.06;173.38;3758;2360;51;51;50.03;47.77;96.74;0.78;52.5;422256;228;-2.66;10.13;0;3778;2360;1877;1.05;62
5;demo2;1633;180.15;58.32;247;37;250;22.42;30.24;23.21;31.16;237.26;6638;2369;42;68;59.16;35.14;98.03;0.36;70.6;294179;194;-0.44;-1.18;0.24;6661;2369;1629;1.12;197
6;demo2;1754;232.87;17.85;247;155;254;18.7;55.2;18.87;56.02;459.07;690;2389;39;103;90.87;24.58;97.73;0.1;103.31;408449;239;-1.63;2.24;3.53;717;2389;1692;1.26;392
(…)
247;demo2;2277;175.8;58.3;248;66;252;20.59;43.12;21.76;41.7;304.58;6697;14845;42;82;77.66;37.33;86.43;0.31;82.1;400304;184;-0.16;-1.52;1.32;6716;14845;2247;1.18;333
248;demo2;1965;188.78;49.94;248;97;252;30.57;31.55;31.69;32.56;371.19;1309;15073;66;64;75.44;33.16;138.42;0.18;85.04;370956;201;-0.23;-1.53;0.92;1318;15073;1947;1.2;370
249;demo2;1986;171.56;58.51;248;64;251;35.05;26.77;36.02;25.47;237.71;4025;16097;73;50;77.36;32.68;26.52;0.44;82.3;340709;182;-0.15;-1.49;0.25;4085;16097;1981;1.12;192
250;demo2;2440;156.6;68.43;247;52;249;32.21;35.05;33.49;37.28;307.55;5472;16389;69;79;87.05;35.69;132.24;0.32;94.07;382116;142;0.07;-1.62;0.12;5481;16389;2437;1.15;201
251;demo2;2846;170.32;67.65;248;58;253;32.21;46.59;31.31;48.64;534.67;1853;16432;57;94;86.1;42.09;84.98;0.13;95.04;484734;180;-0.14;-1.69;1.34;1885;16432;2808;1.25;593

Annexe III : Contenu d'un fichier PID « Learn_xxxx.pid »

LEARNING
!Item;Label;Area;Mean;StdDev;Mode;Min;Max;X;Y;XM;YM;Perim.;BX;BY;Width;Height;Major;Minor;Angle;Circ.;Fer
et;IntDen;Median;Skew;Kurt;%Area;XStart;YStart;Area_exc;Fractal;Skelarea;Ident;Status
101;demo1;2936;204.5;43.47;248;62;251;97.01;32.46;92.15;32.87;517.35;6200;9416;163;61;110.39;33.86;177.59;0.14;16
3.78;600404;218;-1.06;0;0.17;6210;9416;2931;1.2;344;Appendicularians;Learning
102;demo1;6904;155.68;79.5;6;6;252;61.9;141.63;52.69;124.06;781.06;2771;9573;127;222;153.89;57.12;113.54;0.14;225
.94;1074794;173;-0.61;-0.91;0.17;2835;9573;6892;1.2;498;Appendicularians;Learning
103;demo1;5471;190.99;54.03;248;6;252;41.49;164.17;44.21;156.86;812.15;2966;9772;85;253;216.57;32.17;74.97;0.1;25
9.04;1044927;205;-0.9;-0.11;1.26;3036;9772;5402;1.23;683;Appendicularians;Learning
104;demo1;4123;177.51;59.48;248;6;250;44.77;81.43;40.3;92.28;593.9;3628;9876;94;209;158.34;33.15;69.25;0.15;217.1
7;731869;188;-0.65;-0.59;0.15;3706;9876;4117;1.19;311;Appendicularians;Learning
105;demo1;5467;194.65;50.27;247;6;251;105.63;122.77;100.03;114.3;678.09;5439;10083;174;207;229.6;30.32;127.29;0.
15;266.32;1064161;208;-0.94;-0.25;0.55;5439;10083;5437;1.2;422;Appendicularians;Learning
106;demo1;2508;202.18;40.24;248;98;248;47.98;125.19;44.7;120.01;507.53;5957;11567;95;190;148.7;21.47;119.21;0.12;
206.83;507076;216;-0.88;-0.57;0;5964;11567;2508;1.18;229;Appendicularians;Learning
107;demo1;5310;184.58;57.19;182;6;250;125.7;46.99;115.74;51.53;632.61;3905;11836;215;127;189.75;35.63;26.72;0.17;
237.97;980120;197;-1.2;0.74;0.08;4100;11836;5306;1.19;323;Appendicularians;Learning
108;demo1;7624;163.19;74.39;6;6;249;115.15;32.19;140.8;33.51;846.42;6456;11876;344;54;277.3;35.01;177.77;0.13;344
.18;1244141;185;-0.83;-0.49;0.03;6505;11876;7622;1.17;434;Appendicularians;Learning
109;demo1;7046;186.11;55.87;248;6;252;33.91;187.38;36.33;171.78;952.15;2274;12226;102;309;254.52;35.25;78.43;0.1;
318.55;1311328;201;-0.92;-0.05;0.74;2366;12226;6994;1.23;728;Appendicularians;Learning
110;demo1;3689;177.44;56.21;248;6;254;37.59;53.23;39.28;55.38;439.14;5025;12234;114;107;112.33;41.81;130.64;0.24;
145.34;654584;181;-0.66;-0.18;1.92;5049;12234;3618;1.19;302;Appendicularians;Learning
111;demo1;3433;191.05;44.37;248;42;249;68.02;84.03;66.78;91.93;595.75;6368;12576;92;230;161.1;27.13;78.84;0.12;24
3.85;655885;197;-0.56;-0.59;0.06;6441;12576;3431;1.18;247;Appendicularians;Learning
112;demo1;4697;196.41;53.77;248;25;254;116.64;50.51;109.78;53.04;980.95;1049;12855;200;94;179.83;33.26;19.73;0.0
6;208.72;922552;215;-1.01;-0.02;1.36;1218;12855;4633;1.29;1093;Appendicularians;Learning
113;demo1;2978;198.55;45.14;248;67;249;100.47;34.49;95.49;32.94;461.77;5774;12860;168;65;169.14;22.42;161.7;0.18;
177.41;591276;207;-0.99;-0.02;0.2;5774;12860;2972;1.18;278;Appendicularians;Learning
114;demo1;11878;199.91;60.4;248;18;255;64.63;167.89;60.44;154.27;1694.11;1190;13024;126;312;337.81;44.77;107.58;
0.05;326.68;2374527;224;-1.53;1.6;5.57;1196;13024;11216;1.38;2888;Appendicularians;Learning
115;demo1;3948;197.35;52.46;248;7;249;95.13;55.22;89.5;59.66;535.98;4983;13792;161;117;176.57;28.47;37.65;0.17;19
4.25;779123;212;-1.31;0.78;0.18;5131;13792;3941;1.19;361;Appendicularians;Learning
116;demo1;10805;170.22;77.27;6;6;250;98.99;114.53;117.47;136.34;1026.16;5341;15749;259;250;305.61;45.02;132.05;0
.13;345.21;1839236;197;-0.99;-0.28;0.23;5366;15749;10780;1.2;897;Appendicularians;Learning
117;demo1;3922;214.82;46.06;248;51;253;90.31;69.95;85.77;65.88;698.03;2764;15929;161;134;196.3;25.44;138.18;0.1;1
98.21;842537;238;-1.76;2.22;2.04;2782;15929;3842;1.27;791;Appendicularians;Learning
194;demo3;1882;213.13;45.63;248;43;250;47.22;39.63;48.84;38.54;340.86;2802;3756;96;75;97.34;24.62;31.09;0.2;103.2
5;401119;229;-2.02;3.32;0.11;2870;3756;1880;1.18;211;Appendicularians;Learning

Annexe IV : Cahier des charges

Maxime Lédée

Version 1.6
30/05/2013

Institut Français de Recherche pour l'Exploitation de la Mer

Cahier des charges

IUT CALAIS-BOULOGNE

19 rue Louis David
62228 CEDEX Calais

DUT Informatique
Année universitaire : 2012-2013

Note de version

Version	Auteur	Actions
1.0 au 15/04/2013	Maxime Lédée	- Création du document
1.1 au 16/04/2013	Maxime Lédée	- Modification de la partie Présentation du maître d'œuvre - Ajout des modifications de la fonction Learning et Validation dans la liste des besoins - Correction de fautes d'orthographe
1.2 au 30/04/2013	Maxime Lédée	- Modifications des phrases dans les parties II/1. IV/2. - Ordre priorité des besoins 3.6 et 3.7 interchangés
1.3 au 06/05/2013	Maxime Lédée	- MAJ du titre I/ - Ajout parties spécifications techniques et spécifications fonctionnelles - MAJ de la partie VI/
1.4 au 14/05/2013	Maxime Lédée	- Explication du bouton « Keep unsorted thumbnails » détaillée - Explication de la case « Edit tdm file » détaillée
1.5 au 21/05/2013	Maxime Lédée	- Modifications de certains besoins et ajouts de nouveaux besoins suite à la réunion du 17/05/2013
1.6 au 30/05/2013	Maxime Lédée	- Ajout de deux maquettes de la fonction Compilation

Etat :

En cours ☐ Terminé ☐ Validé ■

Table des matières

I/ Objet de la demande 65

II/ Présentation 66
1. L'Ifremer 66
2. Le maître d'ouvrage (MOA) 66
3. Le maître d'œuvre (MOE) 67
4. Dates clefs du projet 67

III/ Description global du projet 68
1. Objectif 68
2. Utilisateurs visés 68

IV/ Spécifications techniques 69
1. Contenu 69
2. Technologie déployée 69
3. Résoudre le problème de gestion de mémoire 69

V/ Spécifications fonctionnelles 71
1. Modification de la fonction VALIDATION 71
2. Modification de la fonction LEARNING 74
3. Modification de la fonction EVALUATION 75
4. Modification de la fonction PREDICTION 76
5. Amélioration à apporter au logiciel Pid Viewer 77
6. Ajouter une nouvelle fonction COMPILATION 77

VI/ Livrables et attendus 79
1. Gestion de projet 79
2. Liste des pièces à livrer au client 79

I/ Objet de la demande

Le *ZooScan* est un appareil permettant d'identifier le zooplancton. La gestion des données issues des identifications est réalisée grâce au logiciel *Plankton Identifier* (*PkId*) développé sous *Delphi 10*. C'est un logiciel libre de droit qui permet l'identification automatique d'objets à partir d'un ensemble d'image et de leurs données associées. Les fichiers lus par *PkId* sont donc des vignettes extraites par le *ZooScan* au format .jpg et les fichiers .pid associés qui contiennent les informations sur la provenance des vignettes, les mesures sur les objets, etc.

Ce logiciel doit évoluer dans ces fonctionnalités et son interface. Le sujet du stage que nous proposons consistera à remodeler l'interface afin qu'elle soit plus simple et plus intuitive, à améliorer les performances du logiciel et à développer de nouvelles fonctionnalités. Le candidat devra posséder de très bonnes qualités d'analyste et de développeur. Pour juger de la bonne marche du projet, un point sera fait régulièrement en présence des responsables thématique et informatique. Pour réaliser les tâches qui lui seront confiées, l'étudiant disposera des sources du logiciel *Plankton Identifier* et de *Borland Developer Studio 2006* ainsi que de documentations techniques et thématiques.

II/ Présentation

1. L'Ifremer

L'**Institut Français de Recherche pour l'Exploitation de la Mer** (**Ifremer** ou **IFREMER**) est un établissement public à caractère industriel et commercial sous la tutelle du ministère de l'Écologie, de l'Énergie, du Développement durable et de la Mer et du ministère de l'Alimentation, de l'Agriculture et de la Pêche. Le site Internet est http://www.ifremer.fr/ .

Au 31/12/12, l'Ifremer c'est :

- ✓ Un budget annuel de 214 millions d'euros (hors opérations internes)
- ✓ 4 navires hauturiers : *Pourquoi pas* ? , *Thalassa* , …
- ✓ 4 navires côtiers : *L'Europe* , *L'Haliotis* , …
- ✓ 6 systèmes sous-marins : *Nautile* , …
- ✓ 1 528 salariés Ifremer et 334 salariés de l'armateur Genavir (gère les navires et les équipements sous-marins de l'Ifremer)
- ✓ 5 centres

Le Centre Manche-Mer du Nord a un périmètre d'intervention qui s'étend de la Belgique à la Baie du Mont Saint-Michel, soient environ 960 kilomètres de côtes et couvrent 6 départements et 4 régions. Ce centre comprend cinq laboratoires dont le Laboratoire Environnement Ressources : c'est dans ce laboratoire que nous avions élaboré le cahier des charges et que je vais effectuer mon stage.

2. Le maître d'ouvrage (MOA)

Les maîtres d'ouvrage est Elvire Antajan, Franck Coppin et Stéphane Gasparini

Le maître d'ouvrage, que l'on appelle également « maîtrise d'ouvrage » ou MOA, est une personne physique ou morale pour laquelle un projet est mis en œuvre et réalisé.

Son rôle : Commanditaire du projet, il définit le cahier des charges et par conséquent les besoins, le budget, le calendrier prévisionnel ainsi que les objectifs à atteindre. Pour réaliser son projet, le maître d'ouvrage fait appel à un maître d'œuvre et peut par ailleurs se faire aider d'une maîtrise d'ouvrage déléguée pour le gérer au mieux. Donneur d'ordre, il suit le projet durant toute sa réalisation, il est le dernier à intervenir si besoin lors de la réception, puisqu'il en est à la fois le pilote et le propriétaire.

3. Le maître d'œuvre (MOE)

Le maître d'œuvre est Maxime Lédée. Son adresse mail est : maxime.ledee [at] ifremer.fr

Le maître d'œuvre est la personne physique ou morale qui est chargé de la réalisation d'un ouvrage.

Son rôle : Le maître d'œuvre conçoit les plans, organise et supervise un projet et livre le produit une fois terminé. Il est choisi par le maître d'ouvrage pour qui le projet est réalisé.

Ses obligations : Le maître d'œuvre se doit donc de respecter les délais fixés au départ, mais aussi le budget qui figure dans le contrat. En contrepartie, il a le choix des moyens techniques qu'il va déployer.

Maxime Lédée établira le planning prévisionnel. Celui-ci comportera toutes les phases de projet.

4. Dates clefs du projet

- ✓ 08 avril 2013 : Début du stage
- ✓ 15 juin 2013 : Fin du stage

III/ Description global du projet

1. Objectif

Rappel : Le *ZooScan* est un appareil permettant d'identifier le zooplancton. La gestion des données issues des identifications est réalisée grâce au logiciel *Plankton Identifier* (*PkId*). Ce logiciel permet l'identification automatique d'objets à partir d'un ensemble d'image et de leurs données associées. Les fichiers lus par *PkId* sont donc des vignettes extraites par le *ZooScan* au format .jpg et les fichiers .pid associés qui contiennent les informations sur la provenance des vignettes, les mesures sur les objets, etc.

PkId comprend quatre fonctions de base :

- ✓ Learning : pour la création du fichier d'apprentissage
- ✓ Evaluation : pour évaluer les performances du fichier d'apprentissage
- ✓ Prédiction : pour l'identification automatique des objets d'un échantillon
- ✓ Validation : pour visualiser et corriger les résultats de la prédiction

L'objectif principal est de mettre à jour le logiciel *Plankton Identifier* en apportant des améliorations de performances, des corrections, et de nouvelles fonctionnalités. L'interface est aussi à améliorer afin qu'elle soit plus simple et plus intuitive.

2. Utilisateurs visés

Les utilisateurs visés sont les suivants :

- ✓ Opérateurs d'Ifremer
- ✓ Partenaires d'Ifremer
- ✓ Opérateurs des autres laboratoires
  - Nationaux
  - Européens
  - Internationaux

IV/ Spécifications techniques

1. Contenu

Le client a fourni au départ :

- ✓ Le code source des deux dernières versions du logiciel *Plankton Identifier*
- ✓ Les exécutables des deux dernières versions du logiciel *Plankton Identifier*
- ✓ L'exécutable de l'environnement de développement intégré (IDE) *Borland Developer Studio 2006*
- ✓ Le manuel d'utilisateur du logiciel *Plankton Identifier*

2. Technologie déployée

Les améliorations et les corrections du logiciel *Plankton Identifier* s'effectuera via l'environnement de développement intégré (IDE) *Borland Developer Studio 2006* et le langage de programmation orienté objet *Delphi*.
Delphi implémente une version orientée objet du langage Pascal renommé *Langage de programmation Delphi*.

Des composants additionnels seront également utilisés :

- ✓ TNTComponents
- ✓ VirtualTreeView
- ✓ CommonLib
- ✓ EasyListView
- ✓ VirtualShellTools
- ✓ VirtualSheelToolsVELVEx
- ✓ Oneinst
- ✓ Delphi Unicode Controls
- ✓ LMD Tools

3. Résoudre le problème de gestion de mémoire

Ce besoin est certainement le plus urgent et est donc à résoudre en première priorité. Les personnes travaillants dans un système win32 sous Windows XP ou Windows 2000 ne peuvent pas travailler avec la version 1.3.0 de *Plankton Identifier*.

Le problème : dans les options Learning et Validation, lorsque l'on est amené à manipuler les vignettes par exemple en les déplaçant ou en modifiant leur taille, le logiciel finit toujours par planter avec le message d'erreur suivant : « ressource système insuffisant »

Remarque : Nous avions pu constater que le problème venait d'une mauvaise gestion du cache des vignettes.

V/ Spécifications fonctionnelles

1. Modification de la fonction VALIDATION

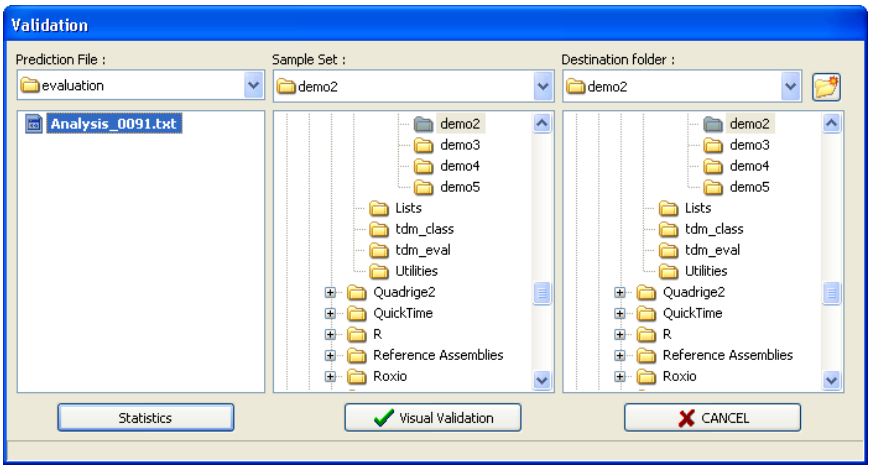
Inputs

Analysis_.txt

- Sans header
- Avec colonne Pred_

Analysis_SampleName.txt

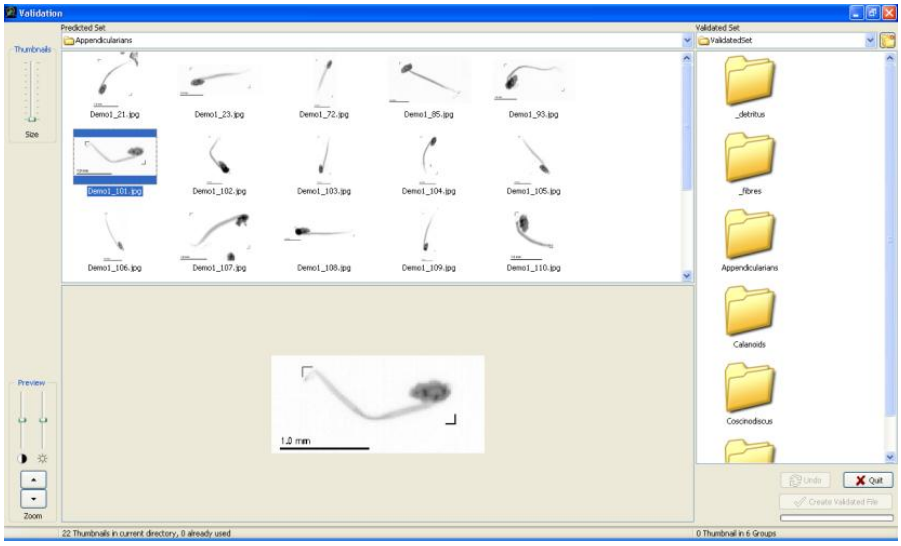
- Avec header
- Avec colonne Pred_



Cette 1ère fenêtre permet de voir le rapport d'analyse de la prédiction et de sélectionner les éléments nécessaires pour trier les vignettes d'un échantillon en fonction de la prédiction.

Modifications :

1- Renommer le bouton 'Statistics' en 'View Analysis Report'
2- Le bouton 'Visual validation' doit simplement trier les vignettes de l'échantillon dans un dossier nommé 'Predicted set', et ne plus créer un dossier 'Validated set' contenant des sous-dossiers vides.
3- Créer sous le bouton 'Visual validation' une checkbox 'Keep unsorted thumbnails' permettant de garder les vignettes non triées ou au contraire de les supprimer. Les maîtres d'ouvrage précisent qu'il faut seulement supprimer les images sous format JPEG qui ont été copiées dans le 'destination folder'.



Outputs

Valid_.txt

- Sans header
- Avec colonne Pred_

Valid_SampleName.txt

- Avec header
- Avec colonne Pred_
- Avec colonne Valid_

Cette 2ième fenêtre permet la correction/validation des vignettes qui ont été automatiquement classées dans le dossier 'Predicted set' en les drag&dropant (!) vers le dossier 'Validated set'. Cette procédure est très longue et fastidieuse, de plus cela crée une nouvelle copie de l'ensemble des vignettes (ça en fait trois en tout : les originales en vrac, les prédites classées selon la prédiction dans Predicted set et les validées classées dans Validated set) ce qui ne présente aucun intérêt en plus de prendre de la place pour rien.

Modifications :

1- Comme on a plus de dossier 'Validated set' cette fenêtre ne doit garder que la partie de droite (qu'il faudra renommer 'Predicted set') avec le preview. La validation se faisant grâce à l'option 'move to' par clic droit de la souris quand une vignette n'a pas été classée dans le bon dossier (comme quand on corrige le Learning set).

2- Petite différence dans les options du clic droit sur une vignette par rapport à celles disponibles dans la fonction Learning, ne pas permettre de supprimer la vignette ! Car en effet toutes les vignettes de l'échantillon doivent être conservées et classées.

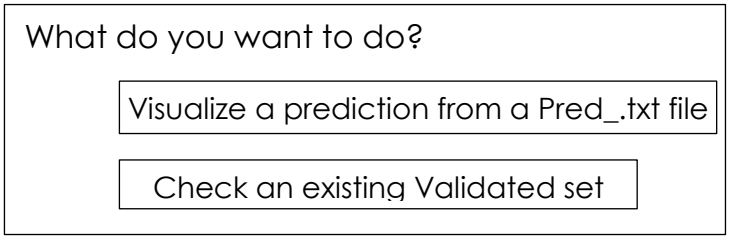
   ↳ **Mise à jour du besoin suite à la réunion du 17/05/2013** : Finalement l'option 'Delete' s'avère nécessaire pour pouvoir supprimer des dossiers qui ne contiendraient plus de vignettes après validation de la prédiction. Il faudrait donc rendre la fonction active lorsqu'on sélectionne un dossier ne contenant pas de fichier .jpeg (mais qui peut contenir un fichier _dat1.pid), et la désactiver lorsqu'on sélectionne une vignette ou un dossier contenant des vignettes.

3- Faire en sorte que l'image affichée en grand lors d'une sélection multiple soit toujours celle de la dernière sélection (actuellement le code affiche la première de la liste ce qui oblige à commencer par la fin).

Nouvelles améliorations établies suite à la réunion du 17/05/2013 :

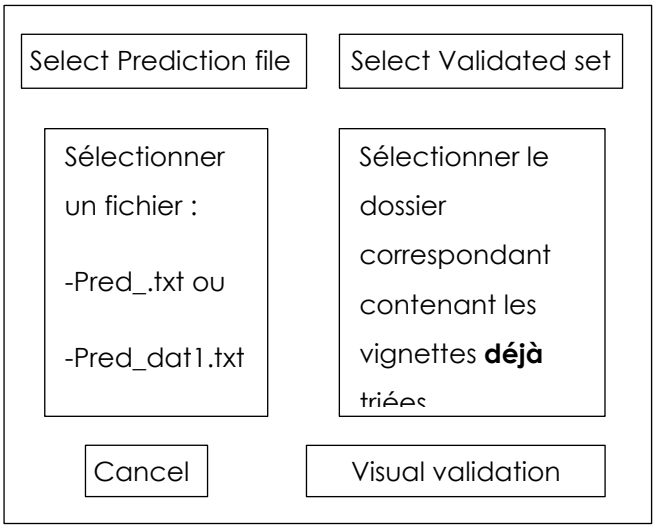
L'usage montre que l'opérateur doit bien souvent revenir sur une validation déjà effectuée (soit pour faire un 2nd contrôle par un autre expert, soit pour faire une identification plus poussée d'un groupe en particulier, etc…). Il faut donc que l'on puisse avoir accès à la deuxième fenêtre de la validation directement depuis la fenêtre principale de *Plankton Identifier*.

1- Lorsqu'on clique sur VALIDATION dans la fenêtre principale de PkId, faire apparaître un message demandant :



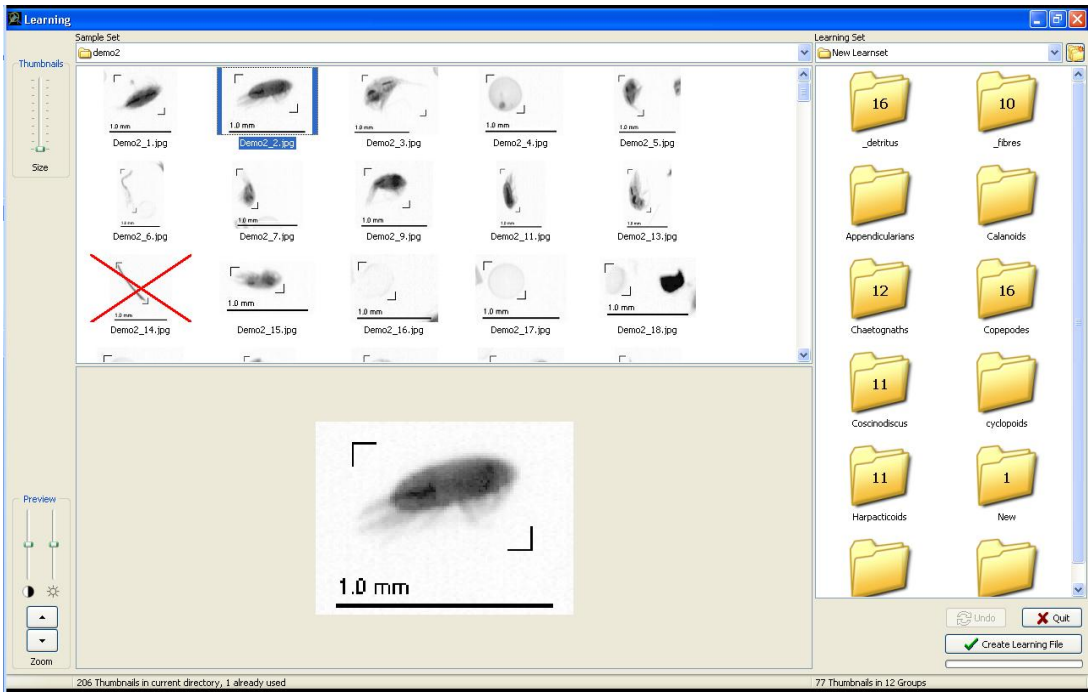
2- Si la première option est sélectionnée, on conserve la procédure actuelle modifiée par Maxime. **Seule différence**, lorsqu'on clic sur 'Visual validation', les vignettes non triées sont copiées et triées directement dans le dossier de destination choisi sans créer de dossier intitulé par défaut 'Predicted set'.

3- Si la deuxième option est sélectionnée, la 1ère fenêtre de validation est modifiée comme suit :



4- Lorsqu'on clique sur 'Visual validation' cela ouvre le dossier de validation choisi dans la 2ième fenêtre crée par Maxime qui reste inchangée.

2. Modification de la fonction LEARNING



Améliorations :

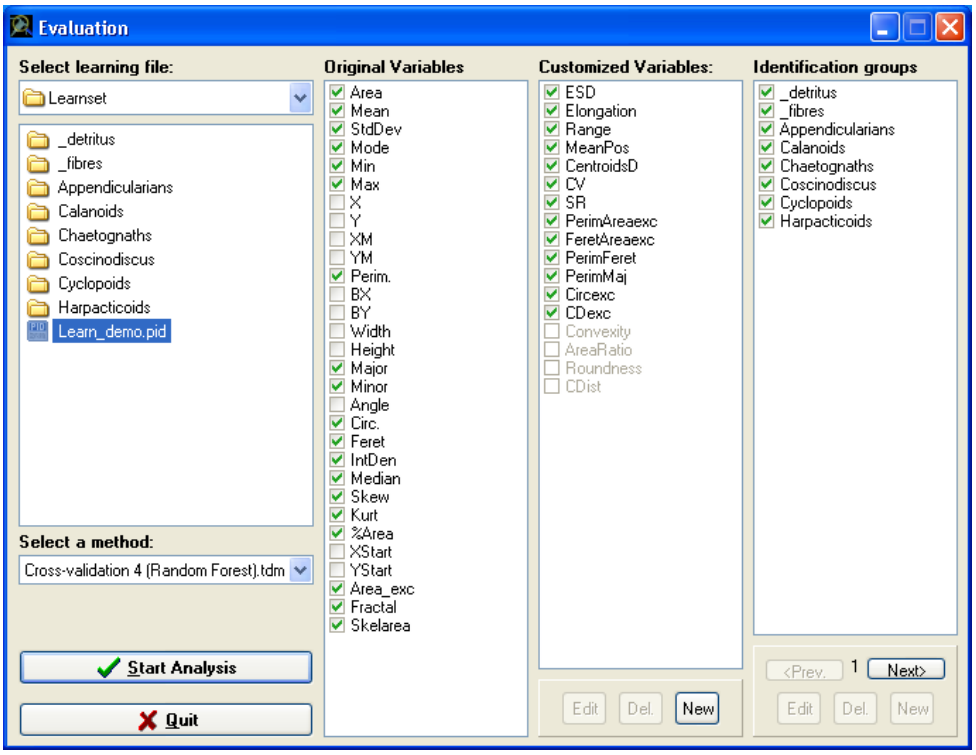
1- Créer un bouton 'dossier parent' dans le volet Sample set (partie de gauche) et dans le volet Learning set (partie de droite), car à l'usage il est assez fastidieux de devoir passer par le dérouleur pour revenir au dossier parent. Ce bouton pourrait être placé à droite du dérouleur (comme c'est déjà le cas pour le bouton 'créer un nouveau dossier' dans le volet Learning Set)

2- Faire que le curseur Thumbnails à gauche pilote à la fois la taille des vignettes ou des dossiers dans Sample set et dans Learning set. Faire que l'on puisse avoir une taille minimale plus petite.

3- Corriger le bug de focus du dossier de destination lors de son survol pour un glisser-déposer (le dossier ne prend pas toujours le focus selon le trajet de la souris, d'où échec si l'on dépose le document)

4- Faire en sorte que l'image affichée en grand lors d'une sélection multiple soit toujours celle de la dernière sélection (actuellement le code affiche la première de la liste ce qui oblige à commencer par la fin).

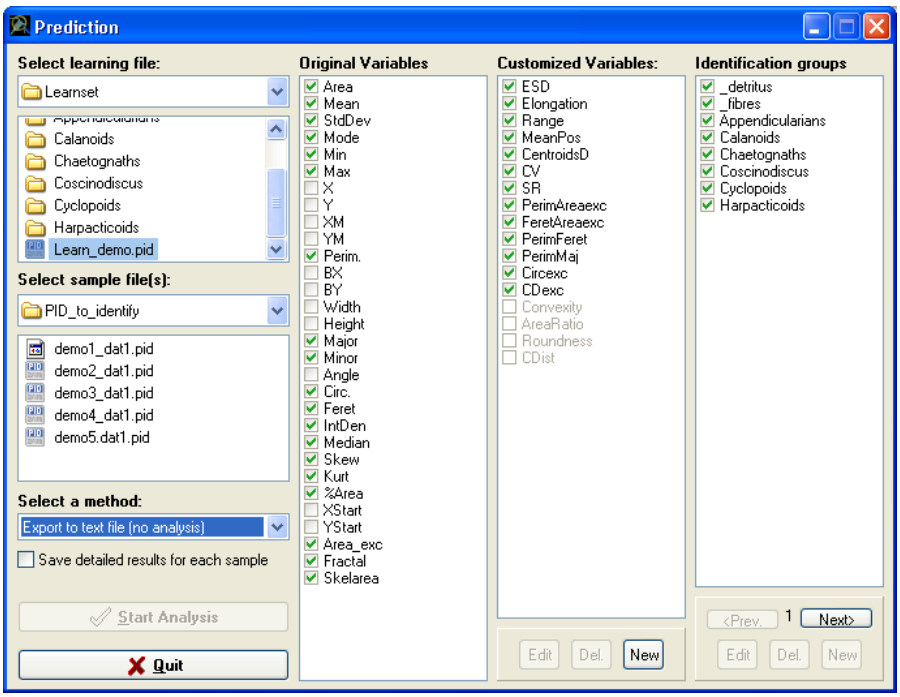
3. Modification de la fonction EVALUATION



Améliorations :

1- Faire qu'à la fin de l'analyse une fenêtre s'ouvre automatiquement pour permettre de visualiser directement le rapport d'analyse html produit par le logiciel Tanagra.
2- Créer un bouton 'Edit tdm file' pour éditer les fichiers tdm et pouvoir les modifier si besoin.
3- Supprimer les checkboxes dans le volet 'Identification groups' (à droite), car on doit toujours utiliser tous les groupes de la liste. **Ce besoin a été classé en basse priorité suite à la réunion du 17/05/2013.**
4- Renommer le bouton 'Start Analysis' en 'Run'.
5- **Nouveau besoin établi suite à la réunion du 17/05/2013** : le fichier de prédiction créé doit commencer par Eval_.txt et non pas Analysis_.txt (à modifier dans la fenêtre enregistrement qui s'ouvre lorsque on clique sur 'Run').

4. Modification de la fonction PREDICTION



Output

Analysis_.txt

- Sans header
- Avec colonne Pred

Analysis_SampleName.txt

- Avec header
- Avec colonne Pred

Améliorations :

1- Faire qu'à la fin de l'analyse une fenêtre s'ouvre automatiquement pour permettre de visualiser directement le rapport d'analyse html produit par le logiciel Tanagra.

2- Créer un bouton 'Edit tdm file' pour éditer les fichiers tdm via le bloc note de Windows et pouvoir les modifier si besoin. L'ouverture des fichiers tdm via bloc note se fait automatiquement lorsque la case 'Edit tdm file' est coché et lorsque l'utilisateur clique sur le bouton 'Run'.

3- Supprimer les checkboxes dans le volet 'Identification groups' (à droite), car on doit toujours utiliser tous les groupes de la liste. **Ce besoin a été classé en basse priorité suite à la réunion du 17/05/2013.**

4- Renommer le bouton 'Start Analysis' en 'Run'.

5- **Nouveau besoin établi suite à la réunion du 17/05/2013** : le fichier de prédiction créé doit commencer par Pred_.txt et non pas Analysis_.txt (à modifier dans la fenêtre enregistrement qui s'ouvre lorsque on clique sur 'Run').

5. Amélioration facultative à apporter au logiciel Pid Viewer

Le logiciel est super pour lire et comparer les fichiers .pid de façon conviviale (en tout cas beaucoup plus qu'avec le bloc-note, y'a pas photo). Néanmoins à l'usage on se rend compte qu'il faut parfois compléter les metadata du fichier .pid si on n'a pas pu le faire lors de l'acquisition de l'image.

Amélioration facultative à faire uniquement s'il reste du temps :

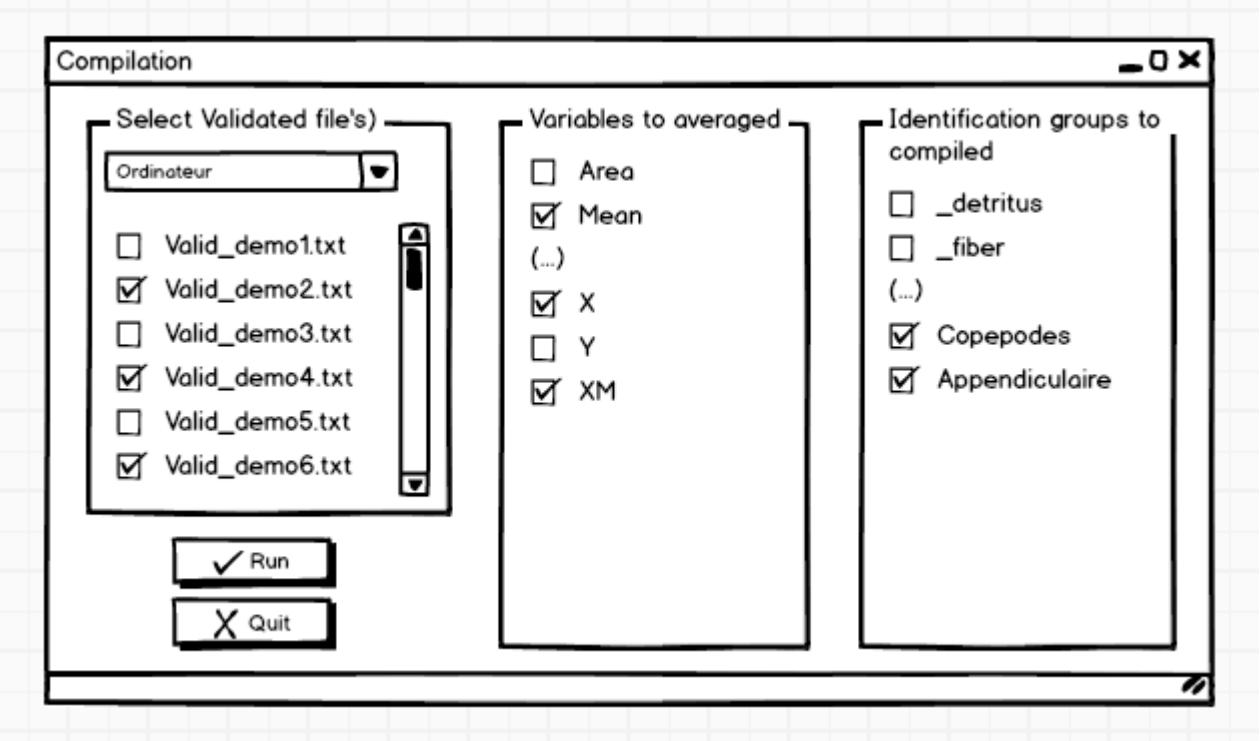
1- Rendre éditable le 'header', mais avec la création d'un fichier .old pour la traçabilité.

6. Ajouter une nouvelle fonction COMPILATION (facultatif, à faire uniquement s'il reste du temps)

Cette nouvelle fonction qui sera accessible dans la fenêtre principale de PkId (à la suite du bouton VALIDATION) permettra d'obtenir un fichier .txt contenant les informations compilées de un ou plusieurs fichiers validés. L'aspect général de la fenêtre correspondante reste à définir.

Mise à jour du 30/05/2013 : ajout des maquettes de la fonction Compilation

1- Maquette de la fenêtre Compilation :



2- Maquette du tableau de résultat de la fonction Compilation présent dans un nouveau fichier créé nommé Compil_SampleName.txt :

Compil_SampleName.txt

SampleName	GroupName	Counts	SubPart	FracId	Area	Mean	...	ESD
Demo1	Copepodes	158	4	...	...	...	...	...
Demo1	Appendiculaires	13	4	..	...	...	...	...
...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...
Demo2	Copepodes	97	16	...	...	...	...	...
Demo2	Appendiculaires	3	16	...	...	...	...	...
...	...	...	...	...	...	...	...	...

VI/ Livrables et attendus

1. Gestion de projet

La nouvelle version doit être prête avant le 15 juin 2013. Un document nommé « Planning prévisionnel » contenant les différentes étapes du projet sera fourni. L'ensemble des livrables produits sera disponible sur le blog http://stagepkid.wordpress.com/ .

2. Liste des pièces à livrer au client

- ✓ La nouvelle version du logiciel *Plankton Identifier* fonctionnelle
- ✓ Le code source
- ✓ Le manuel d'utilisateur (responsabilité d'Elvire Antajan)
- ✓ Les exécutables des composants additionnels à jour

Annexe V : Compte rendu n°3 – Choix de la version d'IDE *Delphi*

Compte rendu n°3 :
Choix de la version d'IDE *Delphi*
10/04/2013
de 16h00 à 16h15 (15min)

- ✓ **Les présents :**
  - Elvire Antajan
  - Stéphane Gasparini
  - Maxime Lédée
- ✓ **Lieu :**
  - Salle de laboratoire du Laboratoire Environnement Ressource à Boulogne sur Mer
- ✓ **Objet de la réunion :**
  - Choisir la version de l'IDE *Delphi* pour apporter les améliorations/corrections du logiciel *Plankton Identifier*
- ✓ **Ordre du jour :**
  1. Problèmes rencontrés
  2. Solution
  3. Version de l'IDE *Delphi* choisie : *Delphi Borland Developer Studio 2006*

---

1. Problèmes rencontrés

Tout d'abord, il faut savoir que la dernière version du logiciel *Plankton Identifier* utilise l'IDE *Delphi Borland Developer Studio 2006 (bds2006)* avec des ajouts complémentaires de composants trouvés sur Internet.

Plusieurs problèmes ont été rencontrés pour choisir la bonne version de l'IDE Delphi :

- ✓ Il était impossible d'installer l'IDE *Delphi Borland Developer Studio 2006* car :
  - Les CD d'installation de *bds2006* sont restés dans le laboratoire d'observation océanographique de Villefranche sur Mer
  - L'exécutable de *Bds2006* est introuvable sur Internet
- ✓ La société *Borland* a été rachetée par la société *embarcadero*
- ✓ La société *embarcadero* propose seulement la dernière version de *Delphi* qui est *Delphi XE3* sorti fin 2012

- ✓ N'ayant pas les CD d'installation de *bds2006*, nous avions tout de même testé *Delphi XE3* en version d'évaluation (40j). Problème : les composants complémentaires n'étaient plus compatibles avec cette nouvelle version
    - Nous avions donc recherché les dernières versions des composants, mais malheureusement les principaux composants n'étaient plus tenus à jour depuis 2006

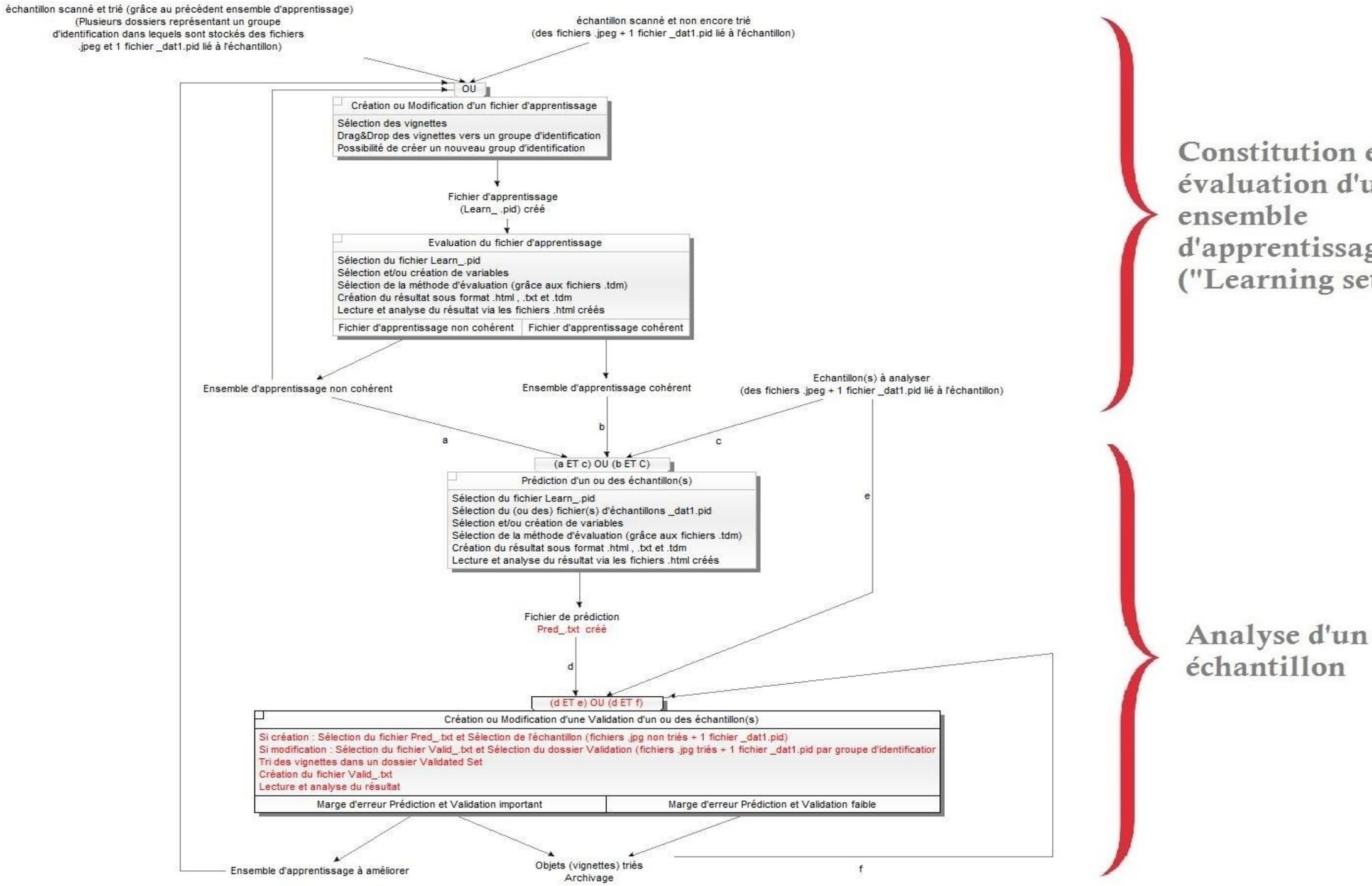
2. Solution

Stéphane Gasparini propose de nous envoyer les CD d'installation de l'IDE *Delphi Borland Developer Studio 2006* par mail et par la poste pour utiliser les mêmes composants complémentaires compatibles avec cette version.

3. Version de l'IDE *Delphi* choisie

La version de l'IDE *Delphi* choisie est donc : *Delphi Borland Developer Studio 2006.*

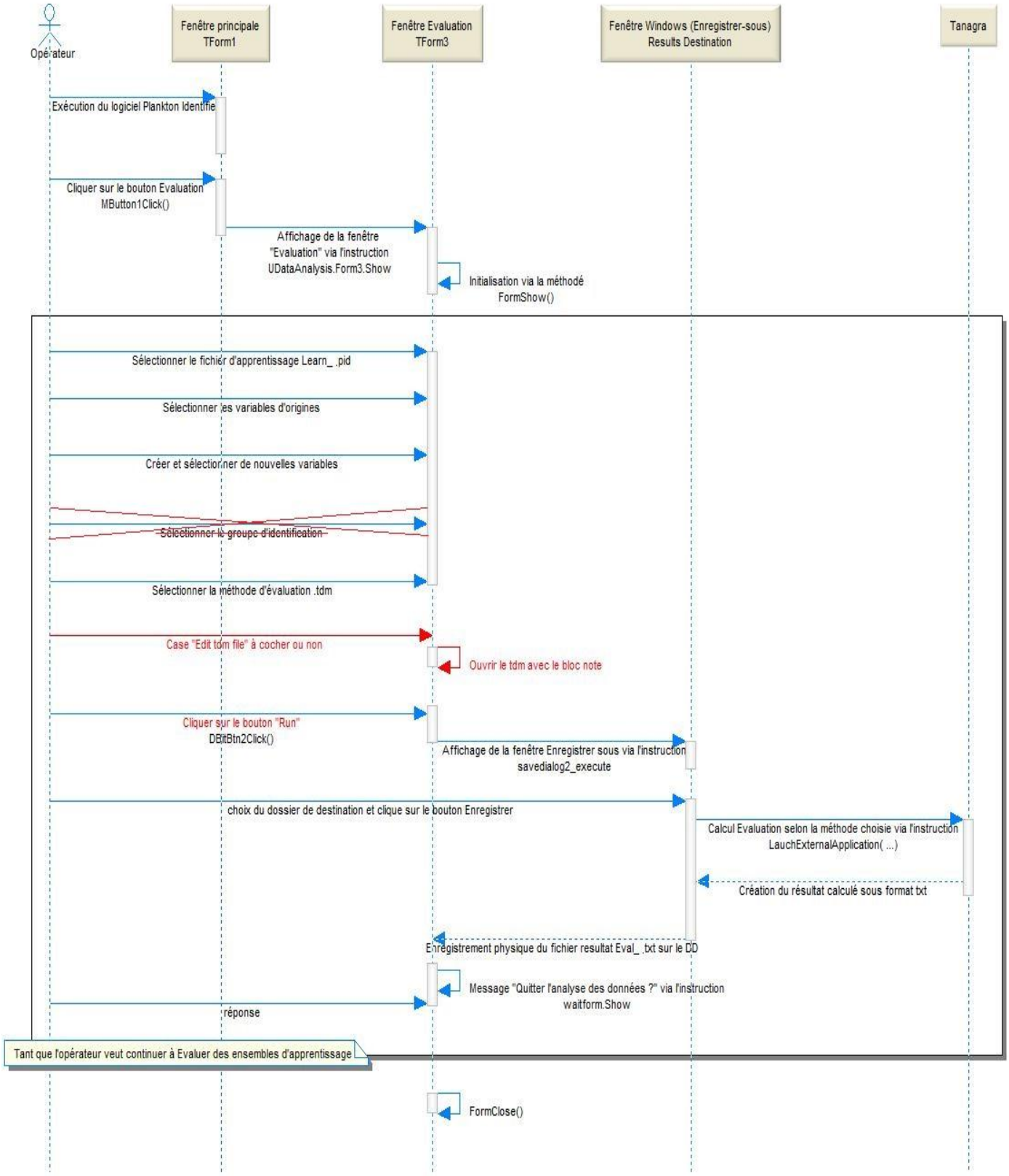
<u>Annexe VI</u> : Modèle Opérationnel de Traitements des fonctionnalités assurées par le logiciel *PLANKTON IDENTIFIER* (après modifications)


échantillon scanné et trié (grâce au précèdent ensemble d'apprentissage)
(Plusieurs dossiers représentant un groupe d'identification dans lequels sont stockés des fichiers .jpeg et 1 fichier _dat1.pid lié à l'échantillon)
échantillon scanné et non encore trié
(des fichiers .jpeg + 1 fichier _dat1.pid lié à l'échantillon)
OU
Création ou Modification d'un fichier d'apprentissage
Sélection des vignettes
Drag&Drop des vignettes vers un groupe d'identification
Possibilité de créer un nouveau group d'identification
Fichier d'apprentissage
(Learn_ .pid) créé
Evaluation du fichier d'apprentissage
Sélection du fichier Learn_.pid
Sélection et/ou création de variables
Sélection de la méthode d'évaluation (grâce aux fichiers .tdm)
Création du résultat sous format .html , .txt et .tdm
Lecture et analyse du résultat via les fichiers .html créés
Fichier d'apprentissage non cohérent
Fichier d'apprentissage cohérent
Ensemble d'apprentissage non cohérent
Ensemble d'apprentissage cohérent
Echantillon(s) à analyser
(des fichiers .jpeg + 1 fichier _dat1.pid lié à l'échantillon)
a
b
c
e
(a ET c) OU (b ET C)
Prédiction d'un ou des échantillon(s)
Sélection du fichier Learn_.pid
Sélection du (ou des) fichier(s) d'échantillons _dat1.pid
Sélection et/ou création de variables
Sélection de la méthode d'évaluation (grâce aux fichiers .tdm)
Création du résultat sous format .html , .txt et .tdm
Lecture et analyse du résultat via les fichiers .html créés
Fichier de prédiction
Pred_.txt créé
d
(d ET e) OU (d ET f)
Création ou Modification d'une Validation d'un ou des échantillon(s)
Si création : Sélection du fichier Pred_.txt et Sélection de l'échantillon (fichiers .jpg non triés + 1 fichier _dat1.pid)
Si modification : Sélection du fichier Valid_.txt et Sélection du dossier Validation (fichiers .jpg triés + 1 fichier _dat1.pid par groupe d'identificatior
Tri des vignettes dans un dossier Validated Set
Création du fichier Valid_.txt
Lecture et analyse du résultat
Marge d'erreur Prédiction et Validation important
Marge d'erreur Prédiction et Validation faible
Ensemble d'apprentissage à améliorer
Objets (vignettes) triés
Archivage
f
Constitution et évaluation d'un ensemble d'apprentissage ("Learning set")
Analyse d'un échantillon

Modèle opérationnel de traitements (MOT) - APRES MAJ		
Logiciel Plankton Identifier : Analyse d'un ou des échantillons		
Auteur : Maxime LEDEE		
Version : 1.6	Créé le : 17/4/2013	Modifié le : 26/5/2013

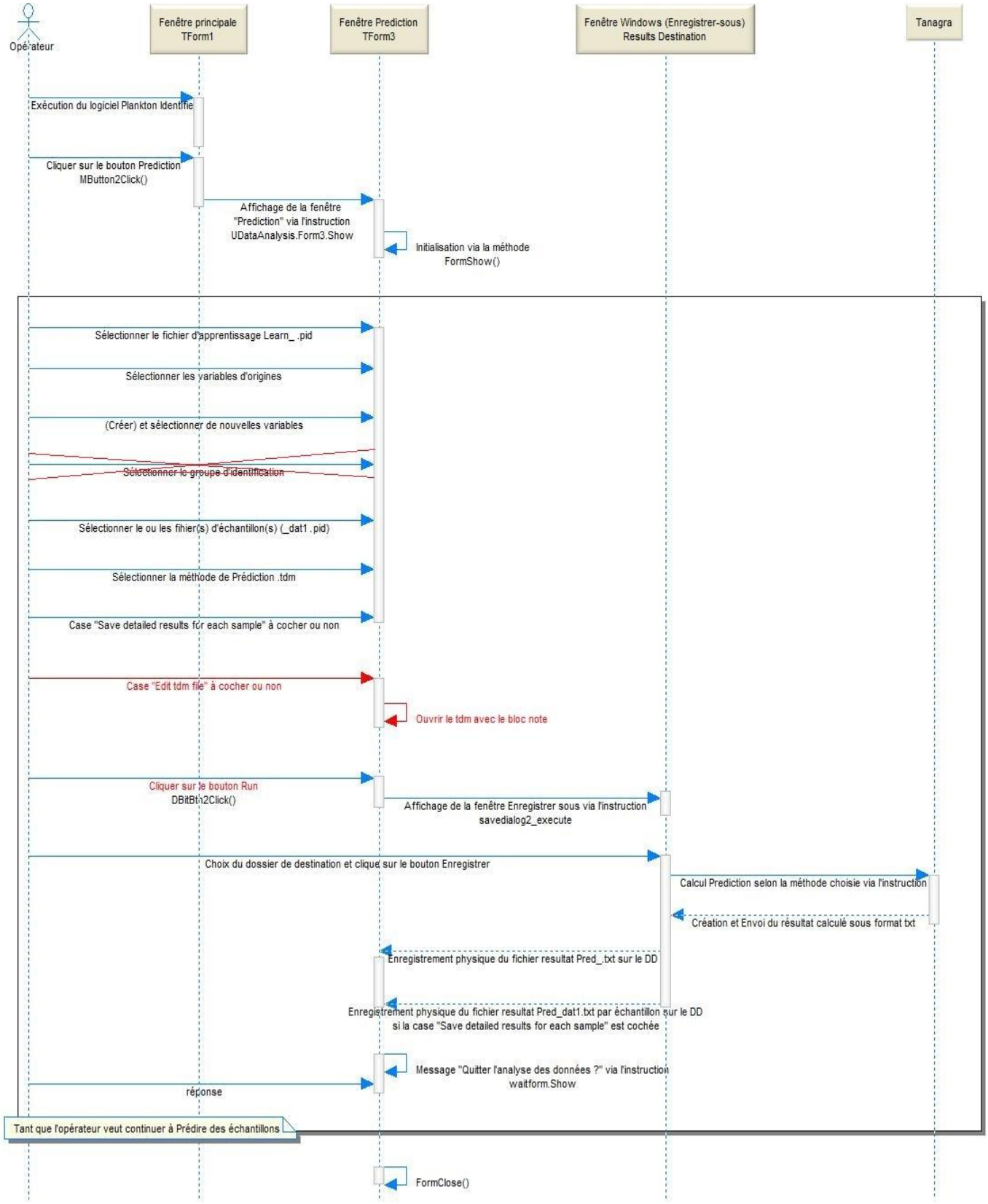
Annexe VII : Diagramme de séquences de la fonction *Evaluation* (après modifications)

Diagramme de séquence après MAJ		
La fonction Evaluation		
Auteur : LEDEE		
Version : 1.3	Créé le : 18/4/2013	Modifié le : 26/5/2013


Opérateur
Fenêtre principale TForm1
Fenêtre Evaluation TForm3
Fenêtre Windows (Enregistrer-sous) Results Destination
Tanagra
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Evaluation MButton1Click()
Affichage de la fenêtre "Evaluation" via l'instruction UDataAnalysis.Form3.Show
Initialisation via la méthodé FormShow()
Sélectionner le fichier d'apprentissage Learn_ .pid
Sélectionner les variables d'origines
Créer et sélectionner de nouvelles variables
Sélectionner le groupe d'identification
Sélectionner la méthode d'évaluation .tdm
Case "Edit tdm file" à cocher ou non
Ouvrir le tdm avec le bloc note
Cliquer sur le bouton "Run" DBitBtn2Click()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog2_execute
choix du dossier de destination et clique sur le bouton Enregistrer
Calcul Evaluation selon la méthode choisie via l'instruction LauchExternalApplication(...)
Création du résultat calculé sous format txt
Enregistrement physique du fichier resultat Eval_ .txt sur le DD
Message "Quitter l'analyse des données ?" via l'instruction waitform.Show
réponse
Tant que l'opérateur veut continuer à Evaluer des ensembles d'apprentissage
FormClose()

Annexe VIII : Diagramme de séquences de la fonction *Prediction* (après modifications)

Diagramme de séquence après MAJ		
La fonction Prediction		
Auteur : LEDEE		
Version : 1.3	Créé le : 18/4/2013	Modifié le : 26/5/2013


Opérateur
Fenêtre principale TForm1
Fenêtre Prediction TForm3
Fenêtre Windows (Enregistrer-sous) Results Destination
Tanagra
Exécution du logiciel Plankton Identifie
Cliquer sur le bouton Prediction MButton2Click()
Affichage de la fenêtre "Prediction" via l'instruction UDataAnalysis.Form3.Show
Initialisation via la méthode FormShow()
Sélectionner le fichier d'apprentissage Learn_ .pid
Sélectionner les variables d'origines
(Créer) et sélectionner de nouvelles variables
Sélectionner le groupe d'identification
Sélectionner le ou les fihier(s) d'échantillon(s) (_dat1 .pid)
Sélectionner la méthode de Prédiction .tdm
Case "Save detailed results for each sample" à cocher ou non
Case "Edit tdm file" à cocher ou non
Ouvrir le tdm avec le bloc note
Cliquer sur le bouton Run DBitBtn2Click()
Affichage de la fenêtre Enregistrer sous via l'instruction savedialog2_execute
Choix du dossier de destination et clique sur le bouton Enregistrer
Calcul Prediction selon la méthode choisie via l'instruction
Création et Envoi du résultat calculé sous format txt
Enregistrement physique du fichier resultat Pred_.txt sur le DD
Enregistrement physique du fichier resultat Pred_dat1.txt par échantillon sur le DD si la case "Save detailed results for each sample" est cochée
Message "Quitter l'analyse des données ?" via l'instruction waitform.Show
réponse
Tant que l'opérateur veut continuer à Prédire des échantillons
FormClose()

Annexe IX : Planning effectif

<table>
<tr><th>09/04</th><th>10/04</th><th colspan="2">11/04</th><th>12/04</th><th>14/04</th><th>15/04</th><th>16/04</th><th>17/04</th><th>18/04</th></tr>
<tr><td>Réunion n°1</td><td>Réunion n°3</td><td colspan="2">Liste des tâches v1.0</td><td></td><td>Liste des tâches v1.1</td><td>Compte-rendu n°3</td><td>Note de cadrage v1.0</td><td>Métamodélisation Delphi v1.0</td><td>Diag. Seq. Learning Avant MAJ v1.0</td></tr>
<tr><td>Réunion n°2</td><td></td><td colspan="2">Diagramme de classe v1.0</td><td></td><td>Charte graphique v1.0</td><td>Installation IDE</td><td>Cahier des charges v1.1</td><td>MCT v1.1</td><td>Diag. Seq. Evaluation Avant MAJ v1.0</td></tr>
<tr><td></td><td></td><td colspan="2">Diagramme de Collaboration v1.0</td><td></td><td>Compte rendu n°1</td><td>Cahier des charges v1.0</td><td>Liste des tâches v1.2</td><td>Actigramme v1.0</td><td>Diag. Seq. Prediction Avant MAJ V1.0</td></tr>
<tr><td></td><td></td><td></td><td>Création du blog</td><td></td><td>Compte rendu n°2</td><td>Plan de communication v1.0</td><td colspan="2">Doc Planning prévisionnel v1.1</td><td></td></tr>
<tr><td></td><td></td><td></td><td>Paramétrages du blog</td><td></td><td></td><td></td><td>Diagramme de PERT v1.0</td><td>Diagramme de GANTT v1.0</td><td></td></tr>
</table>

19/04	…	29/04	30/04	02/05	03/05	06/05
Diag. Seq. Learning : - Avant MAJ V1.1 - Après MAJ V1.1		Ajouts boutons « Dossier Parent » dans fenêtres Learning et Validation	Création new fiche Validation à partir de Learning	Ajout fonction « Move to » dans Validation	Réunion avec la tutrice	Planning v1.2 - liste tâches v1.3 - PERT v1.1 - FANTT v1.1
Diag. Seq. Evaluation : - Avant MAJ V1.1 - Apres MAJ V1.0		Modifications du nom des boutons « View Analysis Report » et « Run »	Modification de la taille minimale des vignettes	Suppression de l'option « Delete » dans Validation	CR n°4	- Diag. Collaboration v1.2 - Actigramme
Diag. Seq. Prediction : - Avant MAJ V1.1 - Apres MAJ V1.0		Création d'une Checkbox « Keep unsorted thumbnails » dans Validation	Réunion de travail sur les différents travaux réalisés depuis le début du stage	Correction de la fonction du zoomage	MCT v1.2.1	Plan action v1.2 Cahier des charges v1.3
Diag. Seq. Validation : - Avant MAJ V1.0 - Apres MAJ V1.0		Le bouton Visual Validation dans Validation ne permet plus de créer un dossier « Validated Set »	Diag. Seq. Evaluation Apres MAJ v1.1		MCD Echantillon v1.0	Note de cadrage v1.1 Plan de communication v1.1
Plan action v1.0		Recherche résolution du problème cache des vignettes	Diag. Seq. Prediction Apres MAJ v1.1		MCD Ensemble apprentissage v1.0	Diag. Seq. Learning : - Avant v1.2 - Après v1.0
			Diag. Seq. Validation Apres MAJ v1.1		MCD Evaluation v1.0	Diag. Seq. Evaluation - Avant v1.2 - Après v1.2
			Cahier des charges v1.2		MCD Prediction v1.0	Diag. Seq. Prediction -Avant v1.2 - Apres v1.2
			Plan action v1.1			Diag. Seq. Validation - Avant v1.1 - Apres v1.2
Recherche et Installation composants additionnels			MAJ fenêtre validation2			

07/05	10/05	13/05	14/05	15/05	16/05	17/05	21/05
MAJ importante de la fiche et unité Validation2	MAJ des 2 maquettes (version « my Balsamiq »)	Choix de la maquette n°1	Cahier des charges v1.4	Test du composant VELVEx	MAJ « Keep unsorted thumbnails »	Création diapo pour réunion MAJ des besoins	MAJ cahier charges v1.5
Proposition de 2 maquettes Validation2 (version « paint »)			Recherche problème dernière sélection vignette	Recherche autres versions VELVEx	MAJ « Edit tdm file »	Réunion avancement stage	MCT v1.3.1
Ajout bouton « Edit tdm File » dans Evaluation et Prediction			Recherche problème gestion du cache des vignettes		Une fenêtre html s'ouvre automatique après analyse Evaluation et Prediction	CR n°5	MAJ code "Keep unsorted Thumbnails"
Ouverture d'un fichier tdm via bloc note					Comparaison code source 1.2.6 et 1.3.0 (+ composants)		Validation : plus de création de Predicted set
					Installation anciens composants		Modification des noms des fichiers résultats : Eval_ .txt Pred_ .txt
					Problème gestion du cache des vignettes résolu		Problème actualisation lors suppression résolu Remplacement composant Treeview
MAJ Fenêtre validation2							

22/05	23/05	24/05	26/05	27/05	28/05	26/05	30/05
MCT Avant v1.4.1 Apres v1.5.1		MCT Avant v1.4.2 Apres v1.5.2	Maquette n°1 Compilation	Correction plantage lors clic de la touche « Suppr »	MAJ Pid Viewer		
MCD Echantillon v1.1	MCD Echantillon v1.2	MCD Prediction : - Avant v1.2 - Apres v1.2	MCT : - Avant v1.5 - Apres v1.6	Fonction Validation terminée	Opt. ReadOnly sur false	Fonction Save	
MCD ensemble apprentissage v1.1	MCD ensemble apprentissage v1.2	MCD Validation v1.0	MCD ensemble apprentissage v1.3		Lecture code		Diag. Seq. Validation Après - Création v1.4 - Re-vérification v1.4
	MCD Evaluation - Avant v1.1 - Apres v1.1		MCD Evaluation : - Avant v1.2 - Apres v1.2				Maquette Compil_ .txt
	Code copie _dat1.pid corrigé lors d'une Move to vers un nouveau groupe		MCD Prediction : - Avant v1.3 - Apres v1.3 MCD Validation v1.1				Cahier des charges v1.6
			Diag. Seq. - Learning Avant v1.3 + Après v1.1 - Evaluation Avant + Après v1.3	Diag. Seq. Validation Après MAJ : - Création v1.3 - Re-vérification v1.3			Planning effectif
			Diag. Seq. - Prediction Avant Après v1.3 - Validation Avant Après v1.2				
MAJ Fonction Validation (Création, Re-vérification)							